信息安全国家重点实验室
SKLOIS
信息安全国家重点实验室
推·荐·用·书

信息安全理论与技术系列丛书

丛书主编：冯登国

安全协议——理论与实践

冯登国 著

清华大学出版社

信息安全理论与技术系列丛书

安全协议——理论与实践

冯登国　著

清华大学出版社

北　京

内 容 简 介

本书共分 4 篇 16 章。系统地介绍安全协议的基本理论、关键技术以及典型应用和实践。主要内容包括密码算法基础知识,可证明安全性、形式化分析、零知识证明、安全多方计算等基础理论与方法,秘密共享、数字签名、身份识别、密钥交换、健忘传输、公平交换等基本安全协议,以及 Kerberos 协议、X.509 协议、IPSec 协议、TLS/SSL 协议、入侵容忍 CA 协议、基于身份的 PKI 协议、可信计算平台远程证明协议等。

本书可供从事信息安全、密码学、计算机、通信、数学等专业的科技人员、硕士和博士研究生参考,也可供高等院校相关专业的师生参考。

图书在版编目( )数据

安全协议:理论与实践/冯登国著. —北京:清华大学出版社,2011.1
(信息安全理论与技术系列丛书)
ISBN 978-7-302-23290-2

Ⅰ. ①安… Ⅱ. ①冯… Ⅲ. ①计算机网络—安全技术—通信协议 Ⅳ. ①TP393.08

中国版本图书馆 CIP 数据核字(2010)第 147667 号

责任编辑:张 民 赵晓宁
责任校对:时翠兰
责任印制:何 芊

出版发行:清华大学出版社
http://www.tup.com.cn
社 总 机:010-62770175
投稿与读者服务:010-62795954,jsjjc@tup.tsinghua.edu.cn
质 量 反 馈:010-62772015,zhiliang@tup.tsinghua.edu.cn
地 址:北京清华大学学研大厦 A 座
邮 编:100084
邮 购:010-62786544
印 刷 者:北京市季蜂印刷有限公司
装 订 者:三河市新茂装订有限公司
经 销:全国新华书店
开 本:185×260 **印 张**:33.5 **字 数**:805 千字
版 次:2011 年 1 月第 1 版 **印 次**:2011 年 1 月第 1 次印刷
印 数:1~3000
定 价:49.00 元

产品编号:038191-01

丛书序

Preface

信息安全已成为国家安全的重要组成部分，也是保障信息社会和信息技术可持续发展的核心基础。信息技术的迅猛发展和深度应用必将带来更多难以解决的信息安全问题，只有掌握了信息安全的科学发展规律，才有可能解决人类社会遇到的各种信息安全问题。但科学规律的掌握非一朝一夕之功，治水、训火、利用核能曾经都经历了漫长的岁月。

无数事实证明，人类是有能力发现规律和认识真理的。今天对信息安全的认识，就经历了一个从保密到保护，又发展到保障的趋于真理的发展过程。信息安全是动态发展的，只有相对安全没有绝对安全，任何人都不能宣称自己对信息安全的认识达到终极。国内外学者已出版了大量的信息安全著作，我和我所领导的团队近10年来也出版了一批信息安全著作，目的是不断提升对信息安全的认识水平。我相信有了这些基础和积累，一定能够推出更高质量和更高认识水平的信息安全著作，也必将为推动我国信息安全理论与技术的创新研究做出实质性贡献。

本丛书的目标是推出系列具有特色和创新的信息安全理论与技术著作，我们的原则是成熟一本出版一本，不求数量，只求质量。希望每一本书都能提升读者对相关领域的认识水平，也希望每一本书都能成为经典范本。

我非常感谢清华大学出版社给我们提供了这样一个大舞台，使我们能够实施我们的计划和理想，我也特别感谢清华大学出版社张民老师的支持和帮助。

限于作者的水平，本丛书难免存在不足之处，敬请读者批评指正。

冯登国

2009年夏于北京

冯登国，中国科学院软件所研究员，博士生导师，教育部高等学校信息安全类专业教学指导委员会副主任委员，国家信息化专家咨询委员会专家，国家863计划信息安全技术主题专家组组长，信息安全国家重点实验室主任，国家计算机网络入侵防范中心主任。

前言

foreword

随着全球信息化程度的日益提高，网络已经成为人类获取信息、沟通交流以及社会生产和生活活动的一种不可或缺的重要载体和手段，信息安全的重要性和紧迫性日益突显。随着金融、能源、交通、电信等重要基础设施对网络的依赖性逐渐增大，信息安全对于社会和经济的影响也越来越大。信息安全问题已经由个人、团体的隐私与机密性问题上升为国家的战略性问题，而安全协议是解决网络安全问题最直接、最有效的手段之一，它可以有效地解决源认证和目标认证、消息的完整性、匿名通信、抗拒绝服务、抗抵赖、授权等一系列重要安全问题。

安全协议是建立在密码算法基础上的一种高互通协议，它运行在计算机网络或分布式系统中，为安全需求的各方提供一系列步骤，借助于密码算法来达到密钥分发、身份认证以及安全地实现网络通信或电子交易等目的。

我领导的安全协议研究团队，从 1995 年便开始安全协议的研究工作，分两个方面展开研究。重点采用理论与实践相结合的技术路线：一方面是理论与技术研究，主要采用以讨论班为主的自由研究模式；另一方面是应用与实践研究，主要采用以工作组为主的集中研究模式。曾得到中国科学院“百人计划”项目、国家自然科学杰出青年基金项目、国家自然科学基金项目和国家 973 计划项目课题的支持，取得了一批高水平的研究成果，培养了一批核心骨干人才，发表了一批高质量学术论文，也在已出版的部分著作中反映了我们的一些研究成果。我一直想把我们在安全协议方面的研究成果和对安全协议的理解写成一本专著与同行分享，以便更多的学者受益。从 1999 年开始动手写作，转眼就是 10 年，在这期间，易稿数次，举办过多次安全协议研讨会，与数名专家进行过交流，调研过包括研究、应用和实践等在内的各种安全需求。理论与技术在不断发展，应用和实践范围在不断扩大和深入，人类对安全协议的认识水平也在不断提高，在这种背景下，要写一本“好”书非常困难，但我还是下决心推出了这本书，试图系统全面地覆盖安全协议的核心内容，并反映我所领导的团队在这一领域的部分前沿成果。因此，本书是作者及其团队长期从事安全协议研究和实践工作的方法和经验的总结，同时也吸收了国内、外现有相关成果中的许多精华。

本书包括 4 篇 16 章，第 1 篇包括 2 章内容，简要介绍了密码算法的分类和一些典型密码算法，以及安全协议的分类、系统模型、安全属性、设计准则、安全缺陷分类和分析方法等。第 2 篇包括 4 章内容，重点介绍了可证明安全性、形式化分析、零知识证明、安全多方计算等基础理论与方法。第 3 篇包括 6 章内容，重点介绍了秘密共享、数字签名、身份识别、密钥交换、健忘传输、公平交换等基本安全协议。第 4 篇包括 4 章内容，重点介绍了 Kerberos 协议、X.509 协议、IPSec 协议、TLS/SSL 协议、入侵容忍 CA 协议、基于身份的 PKI 协议、可

信计算平台远程证明协议等应用安全协议。

本书在写作过程中，在一些内容的讨论和素材的提供方面，得到了信息安全国家重点实验室相关科研、教学人员和博士生们的鼎力相助，他们包括荆继武教授、薛锐研究员、李宏达教授、张振峰研究员、徐海霞副教授、周展飞副教授、姚刚副教授、徐静副研究员以及陈伟东研究员、邓燚副研究员、庄勇博士、路晓明博士、秦宇博士、陈晓峰博士等，在此一并对他们表示衷心的感谢。

本书在写作过程中，得到了国家973计划项目（No. 2007CB311202）和国家自然科学基金项目（No. 60673083）的支持，也得到了清华大学出版社张民老师的大力支持，在此表示衷心的感谢。

由于水平有限，对一些问题的理解和表述或有肤浅之处，诚请读者批评指正。

冯登国

2010年于北京

目录

Contents

第1篇　绪　　论

第2篇　安全协议基础理论与方法

第 3 篇 基础安全协议

第1篇　绪　　论

计算机网络正以惊人的速度向各个领域渗透,成为各领域发展的新源泉,各种现实世界里的组织与系统正在走进网络这个虚拟的世界里,使网络世界变得越来越精彩。与此同时,安全问题也变得越来越突出,也越来越复杂,而安全协议是解决网络安全问题最有效的手段之一,它可以有效地解决源认证和目标认证、消息的完整性、匿名通信、抗拒绝服务、抗抵赖、授权等一些重要的安全问题。

在深入讨论安全协议之前,首先必须澄清以下两个基本问题。

(1) 安全协议和密码算法的基本含义。

(2) 安全协议和密码算法的基本差别与联系。

为了理解安全协议这一概念,首先要了解什么是协议?所谓协议,就是两个或两个以上的参与者采取一系列步骤以完成某项特定的任务。这个定义包含以下3层意思。

① 协议至少需要两个参与者。一个人可以通过执行一系列的步骤来完成一项任务,但它不构成协议。

② 在参与者之间呈现为消息处理和消息交换交替进行的一系列步骤。

③ 通过执行协议必须能够完成某项任务或达成某种共识。

协议和算法这两个概念不尽相同。算法应用于协议中消息处理的环节。对不同的消息处理方式则要求用不同的算法,而对算法的具体化则可定义出不同的协议类型。因此,可以简单地说,安全协议就是在消息处理环节采用了若干密码算法的协议。具体而言,密码算法为传递的消息提供高强度的加、解密操作和其他辅助操作(如杂凑)等,而安全协议是在这些密码算法的基础上为各种安全性需求提供实现方案。

安全协议是建立在密码算法基础上的一种高互通协议,它运行在计算机网络或分布式系统中,为安全需求的各方提供一系列步骤,借助于密码算法来达到密钥分发、身份认证以及安全地实现网络通信或电子交易等目的。

总之,密码算法是安全协议的基础和核心,安全协议是密码算法应用的纽带和桥梁。

值得注意的是,人们(尤其是密码学研究者)通常将安全协议也称为密码协议,但实际上也有些安全协议如实现抗拒绝服务攻击的安全协议并没有采用密码算法来实现,而且安全协议的研究中也包括各种通信协议的安全性研究,采用的研究工具也主要是计算机科学理论与方法,因此,虽然本书中介绍的主要是基于密码算法的安全协议及相关研究工作,但作者还是认为采用安全协议这一术语更好,这也是安全协议研究工作者的基本共识。

本篇将用两章的篇幅简要概述密码算法和安全协议。第1章简要介绍密码算法的分类和一些典型密码算法。第2章简要介绍安全协议的分类、系统模型、安全属性、设计准则、安全缺陷分类和分析方法概述等。

第 1 章　密码算法概述

密码算法是解决网络和信息系统安全的核心技术，也是构建安全协议的重要基础。本章主要介绍密码算法的分类和一些典型密码算法，包括对称密码算法、公钥密码算法、Hash 函数与 MAC 算法等。

1.1　密码算法的分类

密码算法是密码学中最为核心的一个概念。一个密码算法通常被定义为一对数据变换。其中一个变换应用于数据起源项，称为明文，所产生的对应数据项称为密文。而另一个变换应用于密文，恢复出明文。这两个变换分别称为加密变换和解密变换。习惯上，也使用加密和解密这两个术语。

加密变换的输入是明文数据和一个称为加密密钥的独立数据。类似地，解密变换的输入是密文数据和一个称为解密密钥的独立数据。

显然，密码算法可用来提供机密性。明文是没有受到保护的数据。密文可以在一个不信任的环境中传送，因为如果密码算法是安全的，那么任何不知道解密密钥的人都不可能从密文推断出明文。除此之外，密码算法也有其他用处，如可用于安全协议的设计。

密码算法主要有两大类：一类是对称密码算法，有时称为私钥密码算法或秘密密钥密码算法，这类密码算法的基本特征是加密密钥和解密密钥相同或容易相互导出；另一类是非对称密码算法，有时称为公钥密码算法，这类密码算法的基本特征是有两个不对称的密钥而且从一个难以推出另一个。这两类密码算法具有不同的特性，并且以不同的方法来提供安全服务。除此之外，还有其他辅助密码算法，如 Hash 算法（又称 Hash 函数，或杂凑算法/函数，或散列算法/函数）与 MAC 算法等。

1.2　对称密码算法

对称密码算法的基本特征是用于加密和解密的密钥是一样的或者相对容易推出，如图 1.1 所示。

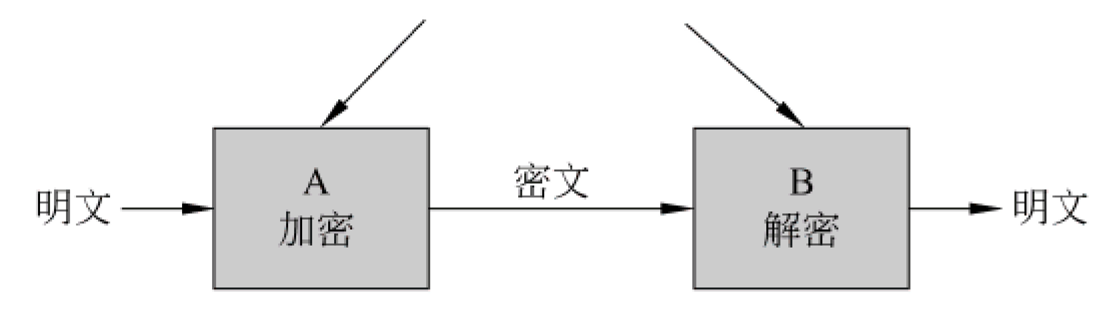


图 1.1　对称密码算法模型

一个对称密码算法的工作流程如下：假定 A 和 B 是两个系统，在它们之间进行秘密通信。它们通过某种方式（物理的或执行某种协议）获得一个共享的秘密密钥，该密钥只有 A

和 B 知道,其他人均不知道。A 或 B 通过使用该密钥加密发送给对方的消息,只有对方可以解密消息,而其他人均无法解密消息。

对称密码算法又分为两种,即序列密码算法(又称流密码算法)和分组密码算法。习惯上,也称为序列(流)密码和分组密码。在序列密码中,将明文消息按字符逐位地进行加密;在分组密码中,将明文消息按固定长度分组(每组含有多个字符),逐组地进行加密。

1.2.1 DES

在 Shannon(香农)提出的密码设计原则下,20 世纪 60～70 年代间,许多研究人员和机构都设计了新的对称密码,其中,由 Feistel 领导的 IBM 公司设计小组设计了 LUCIFFER 密码。在 1973—1974 年间,美国国家标准局(NBS),后来改名为美国国家标准技术研究所(NIST),为了用于保护美国联邦机构的敏感信息,发布了征集加密算法的通告。从提交的建议中,选择了由 IBM 公司提交的 LUCIFFER 算法的改进版本。1975 年,在受控的条件下对此算法作了一些公开的评述。1977 年采纳为联邦信息处理标准 FIPSPUB46,取名为数据加密标准(简称 DES)。该标准很快被用于政府的机密性目的和金融工业界的完整性目的,并且自从那时起,被广泛应用于各个领域。

DES 明文分组和密文分组的长度都为 $L=64$b,轮数 $R=16$,密钥长度 $|\boldsymbol{k}|=64$b,轮密钥长度 $|\boldsymbol{k}_i|=48$b,$i=1,2,\cdots,R$。由于密钥包含 8b 校验位,因此密钥的有效位只有 56b,这就导致自从 DES 公布之日起人们就担心其密钥太短而易受穷搜索攻击,后来这一担心便被证实。当然,DES 的密钥长度可通过使用多重加密来增加。三重 DES 的工作过程如下:首先使用密钥 a 对 64b 的分组加密,然后使用密钥 b 对其加密结果解密,最后再使用密钥 c 加密。通常取 $a=c$。

下面将详细描述 DES。设 m 是一个给定的 64b 明文,c 是 64b 的输出密文,k 是密钥。

1. 密钥方案

每一轮都使用一个不同的、从初始密钥(又称种子密钥)k 导出的 48b 密钥 k_i。k 是一个长度为 64b 的串,第 8b,16b,…,64b 为校验位,共 8 个,这主要是为了检错。在位置 8b,16b,…,64b 是按下述方法给出的:使得 1B(8b)含有奇数个 1。因此在每一个字节中的一个错误都能被检测出。在密钥方案的计算中,不考虑校验位。

密钥方案的计算过程如下:对给定的 64b 的 $\boldsymbol{k}$,删除 8 个校验位,将剩下的用 PC_1 作置换得到 56b 的 C_0D_0,其中 C_0 和 D_0 分别为前、后 28b。对 $1\leqslant i\leqslant 16$ 进行计算,即

$$C_i = \mathrm{LS}_i(C_{i-1}),\quad D_i = \mathrm{LS}_i(D_{i-1}),\quad \boldsymbol{k}_i = \mathrm{PC}_2(C_iD_i)$$

其中,LS_i 表示按轮数不同左循环移位 1 位或 2 位,第 1、2、9、16 轮移动 1 位,其他轮移动 2 位;PC_2 是另一个置换。

置换 PC_1 和置换 PC_2 如图 1.2 所示。

2. 加密过程

(1) 计算 $\boldsymbol{m}_0=\mathrm{IP}(\boldsymbol{m})=L_0R_0$,即通过置换 IP 将一个明文分组 $\boldsymbol{m}=(m_1,m_2,\cdots,m_L)$变换为 $\boldsymbol{m}_0$,这里 L_0 和 R_0 分别表示 $\boldsymbol{m}_0$ 的前(左)、后(右)半部。

(2) 进行 16 轮迭代处理(图 1.3):

$$L_i = R_{i-1},\quad R_i = L_{i-1}\oplus f(R_{i-1},\boldsymbol{k}_i)\quad 1\leqslant i\leqslant 16$$

PC₁						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

PC₂					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	30	36	29	32

图 1.2　置换 PC_1 和置换 PC_2

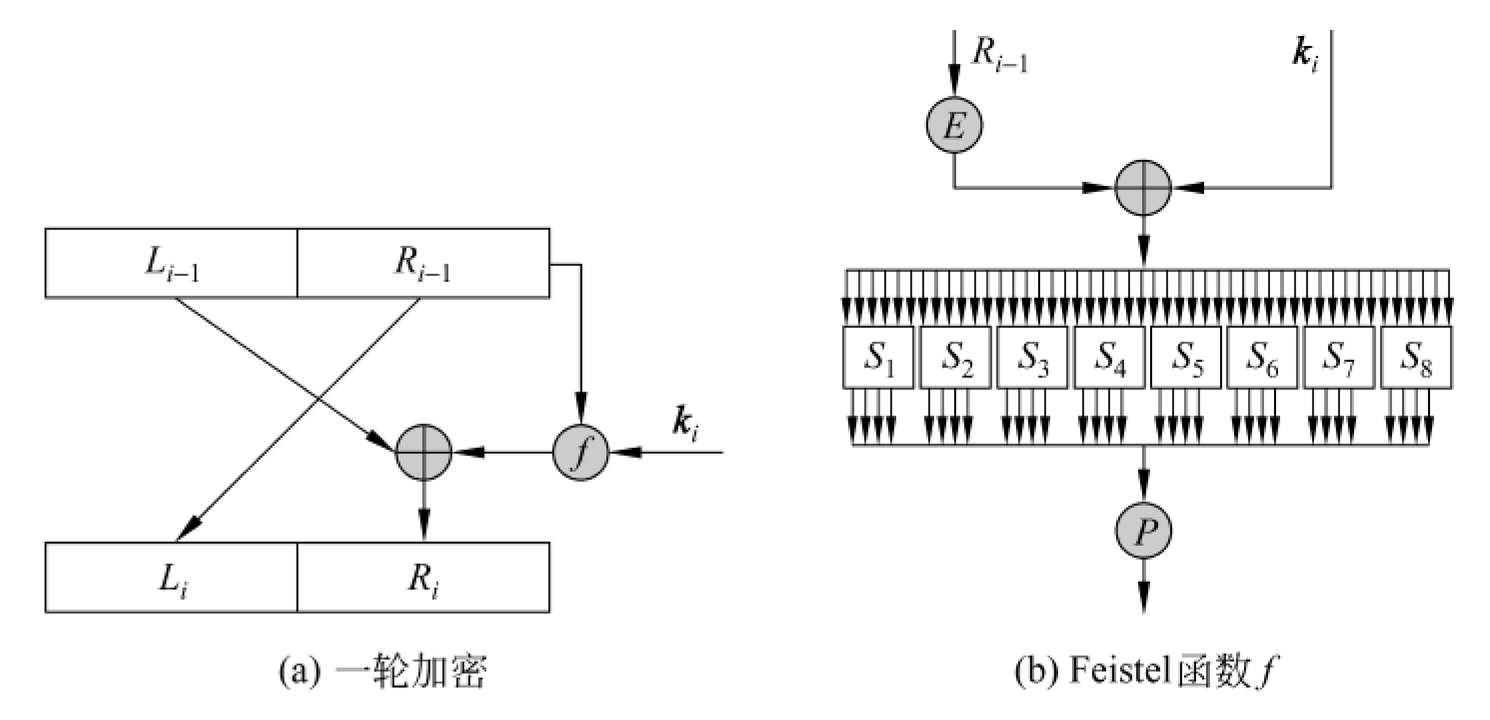


图 1.3　DES 加密迭代过程(E 将 32b 输入扩展为 48b)

以上流程被称为 Feistel 结构,其中⊕表示异或,Feistel 函数 f 先将 $\boldsymbol{k}_i$ 与 R_i 的扩展异或,接着用 8 个 6b 输入、4b 输出的 S 盒和一个置换 P 处理异或后的数据,其中,S 盒是实现非线性映射的常用手段,其输出一般根据输入查表确定。为使加密算法也可用于解密,在第 16 轮迭代后,前、后半部未交换,因此输出实际为 $R_{16}L_{16}$ 而不是 $L_{16}R_{16}$。

(3) 计算 $\boldsymbol{c}=\mathrm{IP}^{-1}(R_{16}L_{16})$,其中 IP^{-1} 表示 IP 的逆。

初始置换 IP 和其逆置换 IP^{-1} 如图 1.4 所示。

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

图 1.4　初始置换 IP 和其逆置换 IP^{-1}

这意味着 m 的第 58b 是 IP(m)的第 1b,m 的第 50b 是 IP(m)的第 2b 等。初始置换 IP 及其逆置换 IP^{-1} 没有密码意义,因为 m 与 IP(m)[或 c 与 $\mathrm{IP}^{-1}(c)$]的一一对应关系是已知

的。它们的作用在于打乱原来输入 m 的 ASCII 码字划分的关系，并将原来明文的检验位 $m_8, m_{16}, \cdots, m_{64}$ 变成 IP 的输出的一个字节。

扩展函数 E 如图 1.5 所示。

置换 P 如图 1.6 所示。

比特选择表 *E*					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

图 1.5　扩展函数 *E*

P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

图 1.6　置换 *P*

3. 解密过程

加密过程中的(3)和(1)的逆是相应的逆置换；在 Feistel 结构下 f 无须可逆，Feistel 结构保证了(2)中各轮处理的可逆，即

$$R_{i-1} = L_i,\quad L_{i-1} = R_i \oplus f(L_i, \boldsymbol{k}_i)\quad (1 \leqslant i \leqslant 16)$$

请读者验证以上加密也能用于解密。

8 个 S 盒如表 1.1 所示。

表 1.1　8 个 *S* 盒

行＼列	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S_1
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	S_2
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	S_3
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	

续表

列/行	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	S_4
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	S_5
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	S_6
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	S_7
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	S_8
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

给定一个长度为 6b 串，比方说 $B_j = b_1b_2b_3b_4b_5b_6$，可按下列办法计算 $S_j(B_j)$：用两个位 b_1b_6 对应的整数 $r(0 \leqslant r \leqslant 3)$ 来确定 S_j 的行（所谓两个位 b_1b_6 对应的整数 r 意指 r 的二进制表示为 b_1b_6，以下的含义类同），用 4 位 $b_2b_3b_4b_5$ 对应的整数 $c(0 \leqslant c \leqslant 15)$ 来确定 S_j 的列，$S_j(B_j)$ 的取值就是 S_j 的第 r 行第 c 列的整数所对应的二进制表示。

1.2.2　AES

1997 年 4 月 15 日，美国国家标准技术研究所（NIST）发起征集高级加密标准（简称 AES）的活动，并专门成立了 AES 工作组，目的是为了确定一个非密级的、全球免费使用的数据加密标准。1997 年 9 月 12 日，在联邦登记处（FR）公布了征集 AES 候选算法的通告。AES 的基本要求是比三重 DES 快而且至少与三重 DES 一样安全，分组长度为 128b，密钥长度为 128/192/256b。1998 年 8 月 20 日，NIST 召开了第一次 AES 候选会议，并公布了 15 个候选算法。1999 年 3 月 22 日，NIST 举行了第二次 AES 候选会议，公布了 15 个候选

算法的讨论结果，并从中选出了5个候选者。2000年4月25日，NIST举行了第三次AES候选会议，对这5个候选算法又进行了讨论。NIST于2000年10月2日公布了最终候选结果，将Rijndael算法作为候选算法。2001年11月26日，NIST将Rijndael算法作为联邦信息处理标准FIPSPUB197对外公布。Rijndael算法的原形是Square算法，它的设计策略是宽轨迹策略(Wide Trail Strategy)，这种策略是针对差分分析和线性分析提出的。Rijndael是一个迭代分组密码，其分组长度和密钥长度都是可变的，但是为了满足AES的要求，限定分组长度为128b，密钥长度为128/192/256b，对应的轮数 r 分别为10/12/14。

下面就来详细描述AES。设 X 是一个给定的128b明文，Y 是128b的输出密文，K 是密钥。

1. 密钥方案

在加密过程中，需要 $r+1$ 个子密钥，需要构作 $4(r+1)$ 个32b字。当种子密钥为128b和192b时，构作 $4(r+1)$ 个32b字的程序是一样的，但当种子密钥为256b时，用另一个不同的程序构作 $4(r+1)$ 个32b字。密钥方案由密钥扩展和轮密钥选择两部分组成。

1) 密钥扩展过程

扩展密钥是一个32b字的数组，且记为 $W[4(r+1)]$，密钥包含在开始的 N_k 个32b字中，其他字由它前面的字经过处理后得到。

当 $N_k=4, r=10$ 或 $N_k=6, r=12$ 时，密钥扩展过程如下。

```
KeyExpansion(CipherKey,W)
    {
      For(i=0;i<Nk;i++) W[i]=CipherKey[i];
      For(j=Nk;j<Nk(r+1);j+=Nk)
      {
       W[j]=W[j-Nk]^S(Rotl(W[j-1]))^RC[j/Nk];
       For(i=1;i<Nk&&i+j<4(r+1);i++)
         W[i+j]=W[i+j-Nk]^W[i+j-1];
      }
    }
```

当 $N_k=8, r=14$ 时，密钥扩展过程如下。

```
KeyExpansion(CipherKey,W)
    {
     For(i=0;i<Nk;i++) W[i]=CipherKey[i];
     For(j=Nk;j<Nk(r+1);j+=Nk)
     {
       W[j]=W[j-Nk]^S(Rotl(W[j-1]))^RC[j/Nk];
       For(i=1;i<4;i++)
         W[i+j]=W[i+j-Nk]^W[i+j-1];
         W[j+4]=W[j+4-Nk]^S(Rotl(W[j+3]));
       For(i=1;i<Nk&& i+j<4(r+1);i++)
         W[i+j]=W[i+j-Nk]^W[i+j-1];
     }
    }
```

其中，Rot1(W)表示对一个字 W 中的字节进行循环置换运算。例如，若输入的一个字的字节为(a,b,c,d)，则产生的输出为(b,c,d,a)。RC[ ]是一个由 8b 轮常数组成的数列，周期为 8，其十六进制表示为：RC[1]=01，RC[2]=02，RC[3]=04，RC[4]=08，RC[5]=10，RC[6]=20，RC[7]=40，RC[8]=80，递归关系式为：RC[i+1]= RC[i]≪1，RC[i]≪1 表示将 RC[i]循环左移一位。“^”表示按位与运算符。

2) 轮密钥选择

轮密钥 K_i 是由密钥缓冲区 $W[4i]$到 $W[4(i+1)]$的字组成。

2. 加密过程

$$Y=O_{K_{r+1}}\circ T\circ \Gamma\circ O_{K_r}\circ \Pi\circ T\circ \Gamma\circ O_{K_{r-1}}\circ \cdots \circ \Pi\circ T\circ \Gamma\circ O_{K_1}(X)$$

其中“∘”表示置换的复合，$K_1,K_2,\cdots,K_{r+1}$是 $r+1$ 个子密钥即由 K 派生的密钥方案。

各个变换如下：

$O_{K_i}:F_2^{128}\mapsto F_2^{128}$ 是一个置换，对 $X\in F_2^{128}$，$O_{K_i}(X)=X\oplus K_i$。

$T:F_2^{128}\mapsto F_2^{128}$ 是一个置换，X 是 T 的输入，首先，把 X 分成 16 个字节，即

$X=(X_{00},X_{01},X_{02},X_{03},X_{10},X_{11},X_{12},X_{13},X_{20},X_{21},X_{22},X_{23},X_{30},X_{31},X_{32},X_{33})$，

输出 $Y=T(X)=(X_{00},X_{01},X_{02},X_{03},X_{13},X_{10},X_{11},X_{12},X_{22},X_{23},X_{20},X_{21},X_{31},X_{32},X_{33},X_{30})$。

$\Pi:F_2^{128}\mapsto F_2^{128}$ 是一个置换，X 是Π 的输入，首先，把 X 分成 16 个字节，即

$X=(X_{00},X_{01},X_{02},X_{03},X_{10},X_{11},X_{12},X_{13},X_{20},X_{21},X_{22},X_{23},X_{30},X_{31},X_{32},X_{33})$，

输出 $Y=\Pi(X)=(Y_{00},Y_{01},Y_{02},Y_{03},Y_{10},Y_{11},Y_{12},Y_{13},Y_{20},Y_{21},Y_{22},Y_{23},Y_{30},Y_{31},Y_{32},Y_{33})$，其中

$$\begin{bmatrix}Y_{0i}\\Y_{1i}\\Y_{2i}\\Y_{3i}\end{bmatrix}=\begin{bmatrix}02&03&01&01\\01&02&03&01\\01&01&02&03\\03&01&01&02\end{bmatrix}\begin{bmatrix}X_{0i}\\X_{1i}\\X_{2i}\\X_{3i}\end{bmatrix}$$

矩阵元素是十六进制数值，如 02 表示比特串 00000010。

$\Gamma:F_2^{128}\mapsto F_2^{128}$ 是一个置换，它由 16 个 F_2^8 上的 S 盒并置构成，$S=L\circ F$，F 是有限域 F_{2^8}[使用不可约多项式 $m(x)=x^8+x^4+x^3+x+1$ 来构作 F_{2^8}，$m(x)$对应的二进制数表示为 100011011，对应的十六进制数表示为“11B”]上的乘法逆，即 $F(X)=X^{-1}$(约定 $F(0)=0$)，$L(X)=AX+b$。

$$\boldsymbol{A}=\begin{bmatrix}1&1&1&1&1&0&0&0\\0&1&1&1&1&1&0&0\\0&0&1&1&1&1&1&0\\0&0&0&1&1&1&1&1\\1&0&0&0&1&1&1&1\\1&1&0&0&0&1&1&1\\1&1&1&0&0&0&1&1\\1&1&1&1&0&0&0&1\end{bmatrix},\quad \boldsymbol{b}=\begin{bmatrix}1\\1\\0\\0\\0\\1\\1\\0\end{bmatrix}。$$

3. 解密过程

解密过程是加密过程的逆，这里不再赘述。

1.2.3 分组密码工作模式

所谓分组密码工作模式是指利用一个基本的分组密码构造出一个密码算法。常用的工作模式主要有以下 4 种。

(1) 电码本(ECB)模式，即直接使用基本的分组密码的模式。给定明文的一个比特串 $x=x_1x_2\cdots$，将它分成一些 bb 长的分组 x_i，b 是分组密码的分组长度，对每一个 x_i 直接使用算法，产生密文分组 c_i。记为 $c_i=e_k(x_i)$，其中 e_k 是加密变换。完整的密文 c 是按次序将 c_i 连接起来，即 $c=c_1c_2\cdots$。其缺点是：在给定密钥的情况下，相同的明文总是产生相同的密文。

(2) 密码分组链(CBC)模式。给定明文的一个比特串 $x=x_1x_2\cdots$，x_i 是 bb 分组。第一个分组 x_1 首先与一个 bb 的初始向量 $\boldsymbol{IV}$ 作异或运算；然后对所得的结果应用加密变换，得到第一个密文分组 c_1。以后的明文分组 x_i 与前一个密文分组 c_{i-1} 作异或运算，对结果再利用加密变换。可按下述方式表示：

$$c_1 = e_k(x_1 \oplus \boldsymbol{IV})$$
$$c_2 = e_k(x_2 \oplus c_1)$$
$$c_i = e_k(x_i \oplus c_{i-1})$$

在该工作模式中，初始向量用于保证由最前面的几个字节相似的明文产生的密文是不同的。

(3) 密码反馈(CFB)模式。先将明文流分成若干个 kb 的字符，$1\leqslant k\leqslant b$，其中 b 表示所用的分组密码的分组长度。每个字符所对应的密文可通过该字符和一个密钥字符相异或获得，该密钥字符通过加密前一个密文并进行截取来获得。在处理过程的开始阶段，将一个 n 比特的初始向量 $\boldsymbol{IV}$ 放在存储密文的地方。对 $k=b$ 的情况，有关步骤可描述如下：算法作用于一个初始化向量 $\boldsymbol{IV}$，产生一个输出 z_1；接着将 x_1 和 z_1 作异或运算产生第一个密文分组 c_1。以后的密文组 c_i 是对前一个密文组使用算法，将输出和对应的明文 x_i 作异或运算，即 $z_1=e_k(\boldsymbol{IV})$，$c_1=x_1\oplus z_1$，$z_2=e_k(c_1)$，$c_2=x_2\oplus z_2$，$\cdots$，$z_i=e_k(c_{i-1})$且 $c_i=x_i\oplus z_i$。

(4) 输出反馈(OFB)模式。该模式与 CFB 模式相似，但它不对密文进行链接，而是通过迭代加密初始向量 $\boldsymbol{IV}$ 来产生密钥字符串。对 $k=b$ 的情况，有关步骤可描述如下：$z_1=e_k(\boldsymbol{IV})$，$c_1=x_1\oplus z_1$，$\cdots$，$z_i=e_k(z_{i-1})$且 $c_i=x_i\oplus z_i$。

工作模式中所使用的初始向量 $\boldsymbol{IV}$ 是一个随机向量，但 $\boldsymbol{IV}$ 未必需要保密，$\boldsymbol{IV}$ 的公开可使人们对消息的起始部分进行攻击。因此，$\boldsymbol{IV}$ 通常要以加密的形式传送，而且要经常变更。

1.2.4 RC4

RC4 是 Rivest 于 1987 年为 RSA 数据安全公司开发的一个密钥长度可变的序列密码。在开始的 7 年中它有专利，算法的细节只有在签署一个保密协议后才能得到。1994 年 9 月，有人把它的源代码匿名张贴到 Cypherpunks 邮件列表中，从而把这个算法泄露了出去。尽管 RSA 公司宣称即使公开代码它仍然是商业秘密，但已为时过晚。

RC4 以 OFB 模式工作，密钥序列与明文相互独立。它有一个 8×8 的 S 盒：$s_0, s_1, \cdots, s_{255}$，该 S 盒是数字 0～255 的一个置换，并且这个置换是一个长度可变的密钥的函数。它有两个计数器：i 和 j，初始值均为 0。

产生一个随机字节的步骤如下：

$$i=(i+1) \bmod 256$$

$$j=(j+s_i) \bmod 256$$

交换 s_i 和 s_j

$$t=(s_i+s_j) \bmod 256$$

$$k=s_t$$

字节 k 与明文异或产生密文或者与密文异或产生明文。加密速度很快，大约比 DES 快 10 倍。

初始化 S 盒的步骤如下：首先进行填充，$s_0=0, s_1=1, \cdots, s_{255}=255$；其次用密钥填充另一个 256B 的数组，以自然的方式重复使用密钥来填充整个数组：$k_0, k_1, \cdots, k_{255}$；最后，将指针 j 设为 0，对 $i=0$～255 完成以下操作：

$$j=(j+s_i+k_i) \bmod 256$$

交换 s_i 和 s_j。

1.3 公钥密码算法

公钥密码算法是由 Diffie 和 Hellman 于 1976 年首次提出的一种密码技术。与对称密码算法相比，公钥密码算法有两个不同的密钥，它可以将加密功能和解密功能分开。一个密钥称作私钥，像在对称密码算法中一样，该密钥被秘密保存。另一个密钥称为公钥，不需要保密。公钥密码算法必须具有以下特性：给定公钥，而要确定出私钥在计算上是不可行的。

公钥密码算法有两种基本的模型：一种是加密模型；另一种是认证模型。参见图 1.7。

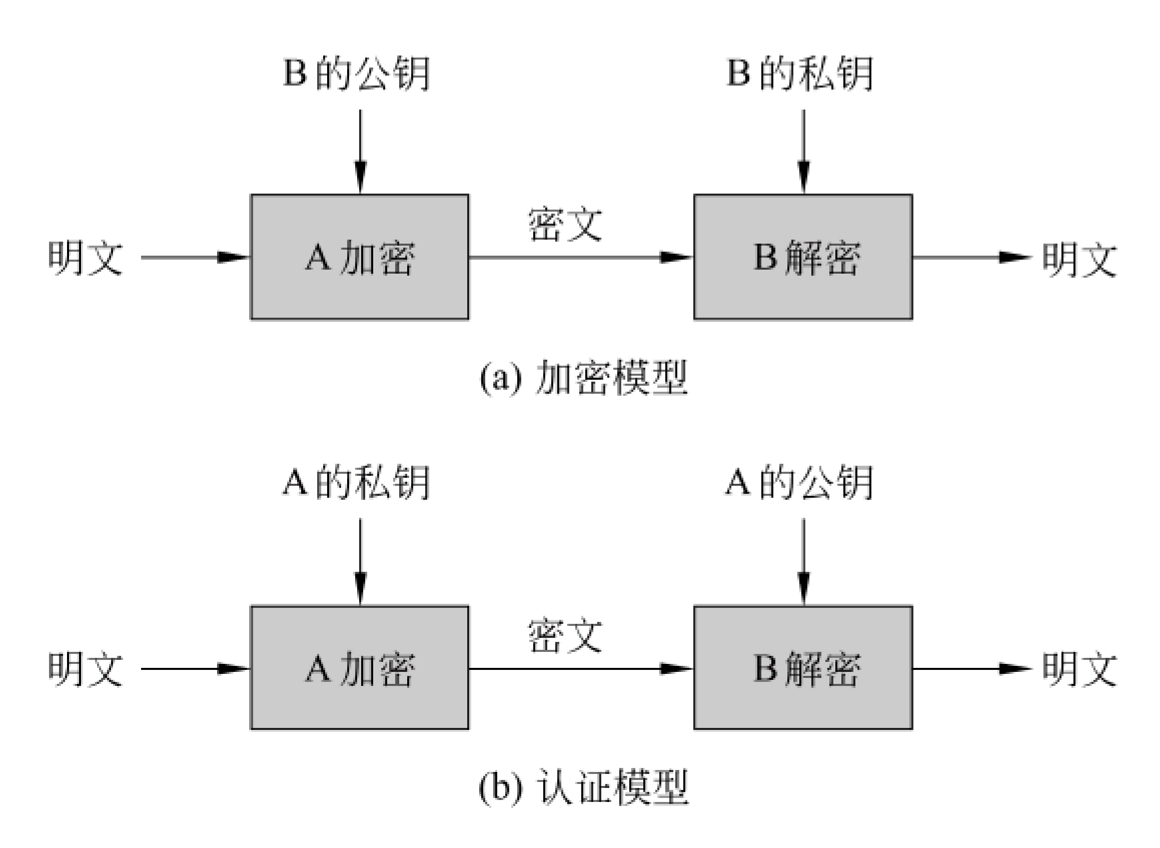


图 1.7 公钥密码算法模型

在加密模型中，假定存在一个包含各通信方的公钥的公开目录（类似于传统的电话号码簿），那么任何一方都可以使用这些密钥向另一方发送机密信息。其具体办法是发送者查出

接收者的公钥并且使用该公钥加密消息。只有拥有相应的私钥的接收者才能解读消息。用于加密模型的算法，通常称之为公钥加密算法。

在认证模型中，将公钥用作解密密钥，任何人均可以从目录中获得解密密钥，从而可解读消息。其具体办法是接收者查出发送者的公钥并且使用该公钥解密消息。只有拥有相应的私钥的人才能产生该消息。可见，公钥密码算法可用于数据起源的认证，并且可确保信息的完整性。

如果一个公钥密码算法可用于这两种模型，则称该算法为可逆公钥密码算法；否则，称为不可逆公钥密码算法。

对算法设计者来说，公钥密码算法的设计比对称密码算法的设计更具挑战性，因为公开的密钥为攻击算法提供了一定的信息。目前所使用的公钥密码算法的安全性基础主要是数学中的难解问题。最流行的有两大类：一类是基于大整数因子分解问题的，如 RSA 算法；另一类是基于离散对数问题的，如 ElGamal 算法、椭圆曲线密码(ECC)算法。

1.3.1 RSA 密码算法

RSA 密码算法是一个可逆的公钥密码算法，以其发明者 Rivest、Shamir 和 Adleman 的首字母来命名。该算法利用了以下基本事实：寻找大素数是相对容易的，而分解两个大素数的乘积在计算上是不可行的。

RSA 密码算法的密钥对的产生过程如下：随机产生两个大素数 p 和 q，计算 $n=pq$，公开 n，而保密 p、q。随机选取一个数 e，e 是小于 $\varphi(n)=(p-1)(q-1)$，且与 $\varphi(n)$ 互素的正整数。利用辗转相除法(也称欧氏算法)，可以找到整数 d 和 r，使得

$$ed+r\varphi(n)=1$$

亦即 $ed\equiv 1 \bmod \varphi(n)$。

数 n、e 和 d 分别称为模、加密指数和解密指数。数 n 和 e 构成公钥，而其余的数 p、q、$\varphi(n)$ 和 d 构成了私钥。

可用数论知识证明，指数 d 和 e 具有以下特征：对任何消息 M，有 $(M^e)^d \bmod n=M \bmod n$。

1. RSA 加密算法

(1) 对消息 M 的加密过程为：$M^e \bmod n$。

(2) 对消息 M' 的解密过程为：$(M')^d \bmod n$。

2. RSA 数字签名算法

(1) 签名生成过程为：签名者首先计算 $C=M^d \bmod n$，将 (M,C) 发送给验证者。

(2) 签名验证过程为：验证者验证是否有 $C^e \bmod n=M$，如果是，签名通过验证。

具体应用时，为了提高效率或破坏算法的某种数学结构(如同态结构)，通常使用 Hash 函数先杂凑要签名的消息，然后对杂凑值进行签名。

1.3.2 ElGamal 型算法

1985 年，ElGamal 提出了一类公钥密码算法，通常称之为 ElGamal 型算法，其安全性基于有限域上计算离散对数的困难性。ElGamal 提出了加密模型和认证模型两种算法。认证模型是美国数字签名算法(DSA)的基础。同年，Koblitz 和 Miller 分别将椭圆曲线用于公钥

密码算法的设计。用椭圆曲线设计的密码算法称之为椭圆曲线密码算法，即 ECC 算法。他们没有发明使用椭圆曲线的密码算法，而是用有限域上的椭圆曲线实现了已存在的公钥密码算法。椭圆曲线上的离散对数的计算要比有限域上的离散对数的计算更困难，能设计出密钥更短的公钥密码算法。因此，成为当前公钥密码算法研究的热点。

1. ElGamal 加密算法

ElGamal 加密算法是非确定性的，因为密文依赖于明文 x 和加密者选择的随机值 k。因此，同样的明文可被加密成许多密文。下面就来描述 Z_p^* 上的 ElGamal 加密算法。

设 p 是一个素数并且离散对数问题在 Z_p 上是计算不可行的，即难处理的，$\alpha \in Z_p^*$ 是一个本原元（也称生成元），即 $Z_p^* = \{\alpha^i, 0 \leqslant i \leqslant p-2\}$，$\beta = \alpha^a \bmod p$，值 p、α 和 β 是公开的，a 是保密的。

（1）对消息 x 的加密过程为：秘密地随机选择一个数 $k \in Z_{p-1}$，计算 $y_1 = \alpha^k \bmod p$，$y_2 = x\beta^k \bmod p$，密文为 (y_1, y_2)。

（2）对消息 x 的解密过程为：对给定的 $y_1, y_2 \in Z_p^*$，计算 $x = y_2(y_1^a)^{-1} \bmod p$。

2. ElGamal 数字签名算法

ElGamal 数字签名算法也是非确定性的，这意味着对任何给定的消息有许多有效的签名，并且验证者能够将它们中的任何一个作为可信的签名而被接受。

设 p 是一个使得在 Z_p 上的离散对数问题是难处理的素数，$\alpha \in Z_p^*$ 是一个本原元，$\beta = \alpha^a \bmod p$，值 p、α 和 β 是公开的，a 是保密的。

（1）签名生成过程为：签名者秘密地随机选择一个数 $k \in Z_{p-1}^*$，计算 $\gamma = \alpha^k \bmod p$，$\delta = (x - a\gamma)k^{-1} \bmod (p-1)$。$(\gamma, \delta)$ 为 x 的签名。

（2）签名验证过程为：验证者验证是否有 $\beta^\gamma \gamma^\delta \equiv \alpha^x (\bmod p)$，如果是，签名通过验证。

3. DSA

DSA 是 ElGamal 数字签名算法的一种变型，它吸收了 Schnorr 数字签名算法的一些思想。DSA 于 1991 年 8 月提出，1994 年 5 月 19 日发表在 Federal Register 上，成为美国的数字签名标准。

设 p 是长 Lb 的素数，在 Z_p 上的离散对数问题是难处理的，其中 $L \equiv 0 (\bmod 64)$ 且 $512 \leqslant L \leqslant 1024$，$q$ 是能被 $p-1$ 整除的 160b 的素数。设 $\alpha \in Z_p^*$ 是 1 模 p 的 q 次根，$\beta = \alpha^a \bmod p$ $(1 \leqslant a \leqslant q-1)$，值 p、q、α 和 β 是公开的，a 是保密的。设 SHA-1 是美国的杂凑算法标准。

（1）签名生成过程为：签名者秘密地随机选择一个数 $k (1 \leqslant k \leqslant q-1)$，计算 $\gamma = (\alpha^k \bmod p) \bmod q$，$\delta = (\text{SHA-1}(x) + a\gamma)k^{-1} \bmod q$（如果 $\gamma = 0$ 或 $\delta = 0$，应该另选一个随机数 k）。(γ, δ) 为 x 的签名。

（2）签名验证过程为：验证者计算 $e_1 = \text{SHA-1}(x)\delta^{-1} \bmod q$，$e_2 = \gamma\delta^{-1} \bmod q$，验证是否有 $(\alpha^{e_1}\beta^{e_2} \bmod p) \bmod q = \gamma$，如果是，签名通过验证。

4. ECDSA

椭圆曲线密码体制中最著名的是 ECDSA，它是将 DSA 算法移植到椭圆曲线上得到的。在一定的假设下可证明其安全性。ECDSA 算法由于安全性高、计算量小、处理速度快和存储空间占用小等优点，越来越受到青睐，已发展成为一个成熟的标准签名算法。

设 T 是一条椭圆曲线 $T=(p,a,b,G,N,h)$，d 是一个随机整数，且满足 $1\leqslant d\leqslant N-1$，$Q=dG$。这里 d 就是私钥，Q 就是公钥。

ECDSA 签名过程如下。

(1) 选择一个随机整数 k，满足 $1\leqslant k\leqslant N-1$。

(2) 计算 $kG=(x_1,y_1)$，并把 x_1 转换成整数$\overline{x_1}$。

(3) 计算 $r=\overline{x_1} \bmod N$，如果 $r=0$，返回第(1)步重新开始。

(4) 计算 $k^{-1} \bmod N$。

(5) 计算 SHA-1(M)，并把位串转换为整数 e。

(6) 计算 $s=k^{-1}(e+dr) \bmod N$，如果 $s=0$，返回第(1)步重新开始。

(7) 签名结果就是(r,s)。

ECDSA 签名验证过程如下。

(1) 验证(r,s)都在区间$[1,N-1]$中。

(2) 计算 SHA-1(M)，并把位串转换为整数 e。

(3) 计算 $w=s^{-1} \bmod N$。

(4) 计算 $u_1=ew \bmod N$，$u_2=rw \bmod N$。

(5) 计算 $X=u_1G+u_2Q$。

(6) 如果 $X=0$，拒绝签名结果，否则把 X 的 x_1 坐标转换成整数$\overline{x_1}$，并计算 $v=\overline{x_1} \bmod N$。

(7) 比较 v 和 r 的值，如果 $v=r$，则签名有效，否则签名验证失败。

1.4 Hash 函数与 MAC 算法

1.4.1 Hash 函数

Hash 函数除了可用于数字签名中提高数字签名的有效性、破坏某些数学结构(如同态结构)和分离保密与签名外，还可用于认证、数据完整性检测、随机数产生和加密。用于完整性检验或数字签名中的 Hash 函数，需具有下列特性。

(1) 函数必须是单向的，即对一个给定的输出(也称消息摘要)，构造一个输入消息将其映射为该输出在计算上是不可行的。

(2) 构造两个不同的消息将它们映射为同一个输出必须是在计算上不可行的。

上述特性中的任何弱点都有可能导致使用 Hash 函数进行完整性检验和数字签名过程中的弱点。例如，如果一个主动攻击者能找到两个不同的消息使得它们的消息摘要相同，则他能用一个消息去替代另一个消息。不管用于产生附件的密码体制的强度如何，这种替代都无法被检测出。

Hash 函数有很多，这里只介绍一个 Hash 函数，即 SHA-1，是美国安全 Hash 标准(SHS)(FIPS PUB 180-2)中的 4 个 Hash 函数之一。美国安全 Hash 标准(SHS)中给出了 4 个 Hash 函数，分别是 SHA-1、SHA-256、SHA-384 和 SHA-512。每个函数的运行都可以分为以下两步进行：预处理和 Hash 计算。预处理阶段包括消息的填充，将填充的消息分为 mb 的等长消息块，Hash 计算中初始值的设置。Hash 计算首先由填充的消息产生一个

消息串，使用消息串，再利用已知的函数、常数及字运算就可以迭代产生一系列的 Hash 值。所得出的最终 Hash 值作为消息摘要。

在描述 SHA-1 之前，引入一些运算符，所有的运算都是在 32b 长的字上进行的。

$\wedge$：按位与运算符。

$\vee$：按位或运算符。

$\oplus$：按位异或运算符。

$\neg$：非运算符。

$+$：模 2^{32} 加法运算。

$\lll$：循环左移运算符。

$\ggg$：循环右移运算符。

SHA-1 的输入消息 x 的长度 $|x|$ 限制为 $<2^{64}$ b，输出长度为 160b，即 5 个字。

给定一个消息 x，$|x|<2^{64}$，首先产生一个长度为 512 倍数的 0、1 串：$M=M_{[0]}M_{[1]}\cdots M_{[N-1]}$，这里 $M_{[i]}(0\leqslant i\leqslant N-1)$都是长为 32b 的串即一个字，$N\equiv 0(\mathrm{mod}\ 16)$。

$M=M_{[0]}M_{[1]}\cdots M_{[N-1]}$的产生办法如下。

(1) 假设消息 x 的长度为 l。首先将比特“1”添加到消息的末尾，再添加 k 个零，这里 k 是方程 $l+1+k\equiv 448 \bmod 512$ 的最小的非负解。然后再添加一个 64b 长的块，其值等于消息 x 的长度 l 的二进制表示。

(2) 将填充后的消息 M 分成 N 个 512b 的块，即 $M=M_{[0]}M_{[1]}\cdots M_{[N-1]}$。

现在从 M 开始构造一个 160b 的消息摘要，其构造过程如下。

(1) 给 5 个寄存器 A、B、C、D、E 赋初始值，$A=67452301$，$B=\mathrm{efcdab89}$，$C=\mathrm{98badcfe}$，$D=10325476$，$E=\mathrm{c2d2e1f0}$。

(2) for $i=0$ to $N/16-1$ do

(3) for $j=0$ to 15 do

$$X_{[j]}=M_{[16i+j]}\text{。}$$

(4) 将给定的 16 个字 $X_{[0]},X_{[1]},\cdots,X_{[15]}$扩展成 80 个字，这一步计算其他 64 个字：

for $t=16$ to 79 do

$$X_{[t]}=(X_{[t-3]}\oplus X_{[t-8]}\oplus X_{[t-14]}\oplus X_{[t-16]})\lll 1\text{。}$$

(5) 将 5 个寄存器 A、B、C、D、E 中的值存储在另外 5 个寄存器 AA、BB、CC、DD、EE 之中，AA$=A$，BB$=B$，CC$=C$，DD$=D$，EE$=E$。

(6) 执行第一轮。

(7) 执行第二轮。

(8) 执行第三轮。

(9) 执行第四轮。

(10) $A=A+\mathrm{AA}$

$B=B+\mathrm{BB}$

$C=C+\mathrm{CC}$

$D=D+\mathrm{DD}$

$E=E+\mathrm{EE}$。

每次处理 16 个字，这 16 个字的处理过程从第(3)步到第(10)步。最后一次，即第 $N/16$

次结束时的寄存器 A、B、C、D、E 的值的级联作为 Hash 值输出，长度为 160b。

下面是 SHA-1 的 4 个轮的描述。

第 1、2、3、4 轮所使用的函数分别如下。

$f_1(X,Y,Z)=(X\wedge Y)\vee(\overline{X}\wedge Z)$。

$f_2(X,Y,Z)=X\oplus Y\oplus Z$。

$f_3(X,Y,Z)=(X\wedge Y)\vee(X\wedge Z)\vee(Y\wedge Z)$。

$f_4(X,Y,Z)=f_2(X,Y,Z)$。

第 1、2、3、4 轮所使用的固定的常数分别如下。

$K_1=5a827999$。

$K_2=6ed9eba1$。

$K_3=8f1bbcdc$。

$K_4=ca62c1d6$。

第 1 轮：

for $k=0$ to 19 do

$\text{TEMP}=(A\ll 5)+f_1(B,C,D)+E+X_{[k]}+K_1$;

$E=D;D=C;C=(B\ll 30);B=A;A=\text{TEMP}$。

第 2 轮：

for $k=20$ to 39 do

$\text{TEMP}=(A\ll 5)+f_2(B,C,D)+E+X_{[k]}+K_2$;

$E=D;D=C;C=(B\ll 30);B=A;A=\text{TEMP}$。

第 3 轮：

for $k=40$ to 59 do

$\text{TEMP}=(A\ll 5)+f_3(B,C,D)+E+X_{[k]}+K_3$;

$E=D;D=C;C=(B\ll 30);B=A;A=\text{TEMP}$。

第 4 轮：

for $k=60$ to 79 do

$\text{TEMP}=(A\ll 5)+f_4(B,C,D)+E+X_{[k]}+K_4$;

$E=D;D=C;C=(B\ll 30);B=A;A=\text{TEMP}$。

1.4.2 MAC 算法

消息认证码(MAC)算法就是满足某种安全属性的带密钥的 Hash 函数。这里介绍两个 MAC 算法，即 HMAC 和 CBC-MAC。

1. HMAC

HMAC 是一种基于 Hash 函数的 MAC，这种方法已经成为国际标准，编号为 ISO/IEC 9797—2:2002。该标准中确定了使用密钥和 Hash 函数或其轮函数计算 MAC 的 3 种方法。第 1 种方法是 MDx-MAC，使用依赖于密钥的方式修改轮函数和设置初始值 IV；第 2 种方法是 HMAC，HMAC 来源于 Internet RFC 2104 和 NIST FIPS 198；第 3 种方法是修改的

MDx-MAC。每一种方法又建议使用 ISO/IEC 10118—3 中的 3 个 Hash 函数中的任何一个，因此 ISO/IEC 9797—2 总共定义了 9 个不同的 MAC 函数。

HMAC 的基本观点是：使用 Hash 函数 H，K_1 和 K_2（$K_1 \neq K_2$）计算 $MAC = H(K_1 || H(K_2 || m))$，其中 K_1 和 K_2 由同一个密钥 K 导出。

HMAC 的工作流程如下。

令 H 是一个 Hash 函数，K 表示密钥，B 表示计算消息摘要时的块按字节的长度（对 SHA-1 是 64B），L 表示消息摘要按字节计算的长度，ipad 表示 0x36 重复 B 次，opad 表示 0x5c 重复 B 次。K 可以有不超过 B 字节的任意长度，但一般建议 K 的长度不小于 L。当使用长度大于 B 的密钥时，先用 H 对密钥进行杂凑，然后用得出的 L 字节作为 HMAC 的真正密钥。

计算一个数据 m 的 HMAC 的操作如下。

(1) 在 K 的后面加上足够的 0 以得到 B 字节的串。

(2) 将第(1)步得到的 B 字节串与 ipad 异或。

(3) 将数据流 m 接在第(2)步得到的 B 字节串后面。

(4) 将 H 应用于第(3)步的比特串。

(5) 将第(1)步所得的 B 字节串与 opad 异或。

(6) 将第(4)步的消息摘要接在第(5)步的 B 字节串后面。

(7) 应用 H 于上一步的比特串。

上面的描述可以表述为 $H((K \oplus \text{opad}) || H(K \oplus \text{ipad} || m))$。

2. CBC-MAC

CBC-MAC 是一种使用密钥和 nb 分组密码计算 mb MAC 的方法，已经成为国际标准，最新编号为 ISO/IEC 9797—1:1999，它是在 ISO/IEC 9797:1994 的基础上增加了一些新内容。

ISO/IEC 9797：1994 陈述了使用一个 nb 分组密码获得 mb MAC（$m \leqslant n$）的过程，其基本思想是：首先，填充数据，形成一串 nb 组；其次，使用 CBC 模式加密这些数据；对最后的输出分组进行选择处理和截断（如果 $m < n$）形成 MAC。

设 nb 数据组是 $D_1, D_2, \cdots, D_q$，则 MAC 的具体计算过程如下。

(1) 置 $I_1 = D_1$，计算 $O_1 = e_K(I_1)$。

(2) 对 $i = 2, 3, \cdots, q$，完成下列计算：$I_i = D_i \oplus O_{i-1}$，$O_i = e_K(I_i)$。

(3) 对 O_q 进行选择处理和截断，获得 mb MAC。

其中 e_K 表示分组密码的加密变换。

MAC 的产生有以下 3 种填充方法。

方法 1：对需要计算 MAC 的数据的右边填充若干个或零个 0b，以便得到一个比特长度是 n 的整数倍的数据串。

方法 2：对需要计算 MAC 的数据的右边先填充一个 1b，然后填充若干个或零个 0b，以便得到一个比特长度是 n 的整数倍的数据串。

方法 3：首先对需要计算 MAC 的数据的右边填充若干个或零个 0b，以便得到一个比特长度是 n 的整数倍的数据串；其次，在所得到的数据串的左边填充一个 nb 组，该组包含了未进行填充的数据的比特长度的二元表示，其左边用 0 补齐。

如果验证者不知道数据的长度，则应选用填充方法 2 或方法 3，因为这两种方法可使验证者查明所填充的那些 0b。

选择处理方法： 如果在计算 CBC-MAC 的过程中没有使用选择处理和截断过程，就会出现安全缺陷。这是因为：假定 $MAC_1 = e_K(D)$，$MAC_2 = e_K(D')$，则 MAC_2 是两个分组消息 $D, D' \oplus MAC_1$ 的一个合法 MAC。这种攻击方法称为截断和粘贴(cut and paste)攻击，可以通过选择处理过程或上述的填充方法 3 来避免。

标准中规定了两种具体的选择处理方法：

(1) 选择一个密钥 K^*，计算 $O_q = e_{K^*}(O_q)$。

(2) 选择一个密钥 K^*，计算 $O_q = e_K(d_{K^*}(O_q))$。

选择处理过程除了避免截断和粘贴攻击之外，还可以增加密码分析者穷搜索密钥 K 的难度。

截断方法： 在选择处理后，取 nb 组的最左边的 mb 构成 MAC。

6 种 CBC-MAC 算法如下。

(1) Algorithm 1＝CBC-MAC with no optional process。

(2) Algorithm 2＝CBC-MAC with optional process as single extra encryption。

(3) Algorithm 3＝CBC-MAC with optional process as extra decryption and encryption (i. e. , triple encrypt last block)。

(4) Algorithm 4＝Double encrypt first and last blocks。

(5) Algorithm 5＝Two parallel instances of algorithm 1 (ex-or outputs)。

(6) Algorithm 6＝Two parallel instances of algorithm 4 (ex-or outputs)。

1.5 密钥管理简介

从前面所描述的可以看到，大部分密码算法都依赖于密钥。密钥的管理本身是一个很复杂的课题，而且是保证安全性的关键点。密钥管理包括确保产生的密钥具有必要的特性，通信双方事先约定密钥的方法以及密钥的保护机制与方法等。密钥管理方法实质上因所使用的密码算法(对称密码算法和公钥密码算法)而异。

所有的密钥都有生存期。所谓一个密钥的生存期是指授权使用该密钥的周期。一般地，一个密钥主要经历以下几个阶段。

(1) 密钥产生，也可能需要登记。

(2) 密钥分发。

(3) 密钥启用/停用。

(4) 密钥替换/更新。

(5) 密钥撤销。

(6) 密钥销毁。

密钥产生必须要考虑具体密码算法的公认的限制，如避免 RSA 的弱密钥。密钥产生也必须确保使用一个合适的随机过程。如果密钥选择过程有一定的约束，就缩小了攻击者搜索密钥的空间。为此，应该用一个适当的随机数生成器来产生密钥。最好的办法是使用物理噪声源。当然，也可以使用一个用秘密的随机初始种子控制的伪随机软件过程来产生密

钥,但是必须慎重使用。

密钥登记包括将产生的密钥与特定的使用捆绑在一起。例如,用于数字签名的密钥,必须与签名者的身份捆绑在一起。这个捆绑必须通过某一授权机构来完成。

密钥分发包括对称密码算法的密钥分发和公钥密码算法的密钥分发,二者的要求有着本质的差别。在一个对称密码算法中,要求将一个密钥从通信的一方通过某种方式发送到另一方,只有通信双方知道密钥而其余任何一方都不知道该密钥。在一个公钥密码算法中,要求私钥只有通信一方知道,而其余任何一方都不知道,与私钥匹配使用的公钥是公开的,任何人都可以使用该公钥和私钥的拥有者进行秘密通信。当分发一个公钥时,不需要机密性。然而,公钥的完整性是必需的。绝对不允许攻击者用别的值替代成员 A 的公钥使得成员 B 相信该值是成员 A 的公钥。因此,公钥分发不像在电话簿上公布电话号码那样简单,它需要以某种特定的方式来分发。目前人们主要采用证书的方式来分发公钥。一个公钥证书是一个数据结构,它将某一成员的标识符和一个公钥值捆绑在一起。证书数据结构由某一被称为认证机构的成员进行数字签名。

密钥启用/停用和密钥替换/更新与密钥分发有关,也是密钥管理中很重要的环节。

密钥撤销在特定的环境中是必需的。原因是密钥撤销包括与密钥有关的系统的迁移,怀疑一个特定的密钥已受到威胁或密钥的使用目的已经改变(如提高安全级别)。

密钥销毁包括清除一个密钥的所有踪迹。一个密钥的值在被停止使用后可能还要持续一段时间,如一条记载的加密数据流包含的信息可能仍然需要保密一段时间。为此,使用的任何密钥的秘密性都需要保持到所保护的信息不再需要保密为止。这表明在密钥的使用活动终结后,安全地销毁所有敏感密钥的副本是十分重要的。例如,必须使得一个攻击者通过观察旧的数据文件,存储的内容或抛弃的设备,确定旧密钥值是绝对不可能的。

一般地,密钥从产生到终结的整个生存期中,都需要加强保护。所有密钥的完整性也需要保护,因为一个入侵者可能修改或替代密钥,从而危及机密性服务。另外,除了公钥密码算法中的公钥外,所有的密钥需要保密。在实际中,存储密钥最安全的方法是将其放在物理上安全的地方。当一个密钥无法用物理的办法进行安全保护时,如当密钥需要从一个地方传送到另一个地方时,密钥必须用其他的方法来保护,诸如:

(1) 由一个可信方来分发;

(2) 将一个密钥分成两部分,委托给两个不同的人或机构;

(3) 通过机密性(如用另一个密钥加密)和完整性服务来保护。

在现代信息系统中,密钥分发主要通过密钥分配和密钥协商等密钥交换协议来完成,这些协议的安全性研究非常重要,而且已将这些研究内容都归结为安全协议的研究范围。

1.6　小结

本章简要介绍了一些典型的密码算法,其真正目的是为了更好地描述安全协议或理解安全协议,因此,没有对其安全现状及其他方面做阐述。对密码算法感兴趣的读者可参阅文献[1]～[5],想了解密码学最新进展的读者可参阅文献[6]～[8]。

参 考 文 献

[1] 冯登国,裴定一.密码学导引.北京:科学出版社,1999.
[2] Stinson,D R. Cryptography: Theory and Practice,Second Edition,CRC press,2002(密码学原理与实践.第 2 版,冯登国译.北京:电子工业出版社,2003).
[3] 冯登国,林东岱,吴文玲.欧洲信息安全算法工程.北京:科学出版社,2003.
[4] Menezes A J,Van Oorschot P, Vanstone S. Handbook of Applied Cryptography, CRC press,1996(应用密码学手册. 胡磊,王鹏译,北京:电子工业出版社,2005).
[5] Schneier B. Applied Cryptography: Protocols,Algorithms,and Source Code in C,John Wiley & Sons,1994(应用密码学——协议、算法和 C 源程序. 吴世忠,祝世雄,张文政等译.北京:机械工业出版社,2004).
[6] 中国密码学会组编. 中国密码学发展报告 2007. 北京:电子工业出版社,2008.
[7] 中国密码学会组编. 中国密码学发展报告 2008. 北京:电子工业出版社,2009.
[8] 中国科学技术协会主编,中国密码学会编著. 密码学学科发展报告. 北京:中国科学技术出版社,2010.

第2章　安全协议概述

本章主要介绍安全协议的分类、系统模型、安全属性、设计准则、安全缺陷分类、基础理论与方法概述等。

2.1　安全协议的分类

安全协议的分类问题还没有一个被学术界广泛认可的一般方法，一般来说，看问题的角度不同，分类就不同。本节从作者的视角介绍了安全协议的分类问题。

从安全协议所要实现的目的来看，可以将现有的最常用的安全协议分为以下6类。

(1) 密钥交换协议。该类协议用于完成会话密钥的建立。一般情况下是在参与协议的两个或者多个实体之间建立共享的秘密，如用于一次通信中的会话密钥。协议中的密码算法可采用对称密码算法，也可采用公钥密码算法。

已有很多密钥交换协议，如Diffie-Hellman协议、Blom协议、MQV协议、端-端协议、MTI协议和Girault协议等。

(2) 认证协议。该类协议用于防止假冒、篡改、否认等攻击，实现身份认证、消息完整性认证、数据源和目标认证等，主要包括实体认证(身份认证，又称身份识别)协议和数字签名协议。

最有代表性的身份认证协议有两类：一类是1984年由Shamir提出的基于身份的身份认证协议；另一类是1986年Fiat等人提出的零知识身份认证协议。随后，人们在这两类协议的基础上又提出了一系列实用的身份认证协议，如Schnorr协议、Okamoto协议、Guillou-Quisquater协议和Feige-Fiat-Shamir协议等。

数字签名协议主要有两类：一类是普通数字签名协议，该类数字签名协议通常称为数字签名算法，如第1章中介绍的RSA数字签名算法、DSA；另一类是特殊数字签名协议，如不可否认的数字签名协议、Fail-Stop数字签名协议、群数字签名协议等。

(3) 认证密钥交换协议。该类协议将认证协议和密钥交换协议结合在一起，先对通信实体的身份进行认证，在认证成功的基础上，为下一步安全通信分发所使用的会话密钥。

常见的认证密钥交换协议有互联网密钥交换(IKE)协议、分布式认证安全服务(DASS)协议、Kerberos协议、X.509协议等。

当然，这类协议既可纳入密钥交换协议，又可纳入认证协议。

(4) 安全电子支付和安全电子交易协议。该类协议用于电子商务系统中以确保电子支付和电子交易的安全性、可靠性和公平性。电子商务系统中交易的双方，往往其利益目标不一致。因此，该类协议最为关注的就是公平性，即协议应保证交易双方都不能通过损害对方利益而得到它不应得的利益。

常见的安全电子支付和电子交易协议有SET协议、iKP协议和电子现金等。

(5) 安全通信协议。该类协议用于计算机通信网络中以确保信息的安全交换等。

常见的安全通信协议有 PPTP/L2PP 协议、IPSec 协议、SSL/TLS 协议、PGP 协议、S/MIME 协议、S-HTTP 协议和 SNMPv3 协议等。

(6) 安全多方计算协议。该类协议的目的是保证分布式环境中各参与方以安全的方式来共同执行分布式的计算任务。该类协议的两个最基本的安全要求是保证协议的正确性和各参与方私有输入的秘密性,即协议执行完之后每个参与方都应该得到正确的输出,并且除此之外不能获知其他任何信息。该类协议的实例包括秘密共享、掷币(Coin-tossing)、安全广播、网上选举、电子投标和拍卖、合同签署、匿名交易、保密信息检索、保密数据库访问、联合签名和联合解密等协议。

用高度抽象的观点来看,可以将大部分已有的安全协议归纳为安全多方计算协议,但这样做一般不利于问题的解决。

从安全协议与具体应用的关联性来看,可以将现有的最常用的安全协议分为以下两类。

(1) 基础安全协议。该类协议与具体应用无关,是设计应用安全协议或其他复杂协议的基础,如秘密共享协议、数字签名协议、身份识别协议、密钥交换协议、健忘传输协议和公平交换协议等。

(2) 应用安全协议。该类协议与具体应用有关,是用基础安全协议或密码算法结合具体应用构建的协议,如 Kerberos 认证协议、X. 509 协议、IPSec 协议、TLS/SSL 协议、SET 协议、PKI/CA 协议和可信计算协议等。

本书将按照第 2 种分类方式介绍相关安全协议。

2.2 安全协议系统模型

在一个大的分布式环境中运行安全协议所面临的最大问题是,其所处的网络通信环境是不安全的。如果将协议及其所处的环境视为一个系统,那么在这个系统中,一般而言包括发送和接收消息的诚实主体和一个攻击者,以及用于管理消息发送和接收的规则。协议的合法消息可被攻击者截取、修改、重放、删除和插入。攻击者将所有已知的消息放入其知识集合 KS(Knowledge Set)中。诚实主体之间交换的任何消息都将被加入到攻击者的 KS 中,并且攻击者可对 KS 中的消息进行操作,将所得消息也加入到 KS 中。攻击者可进行的操作至少包括级联、分离、加密和解密。图 2.1 是安全协议系统模型的示意图。

一个被动攻击者可在线窃听敏感信息。而一个主动攻击者则可截获数据包并对其进行任意的修改,甚至可以伪装成通信主体欺骗诚实主体与其进行非法通信。加密运算可以有效地阻止主动入侵,因为在不知道密钥的前提下,对密文消息的丝毫改动都将导致解密运算的失败,此时攻击者能做的仅仅是阻止消息送达或准时送达其目的地。归纳起来,攻击者的行为表现为以下几种形式。

(1) 将消息发送到其意定接收者。

(2) 延迟消息的送达。

(3) 将消息修改后转发。

(4) 将消息与以前接收的消息合并。

(5) 改变部分或全部消息的目的地址。

(6) 重放消息。

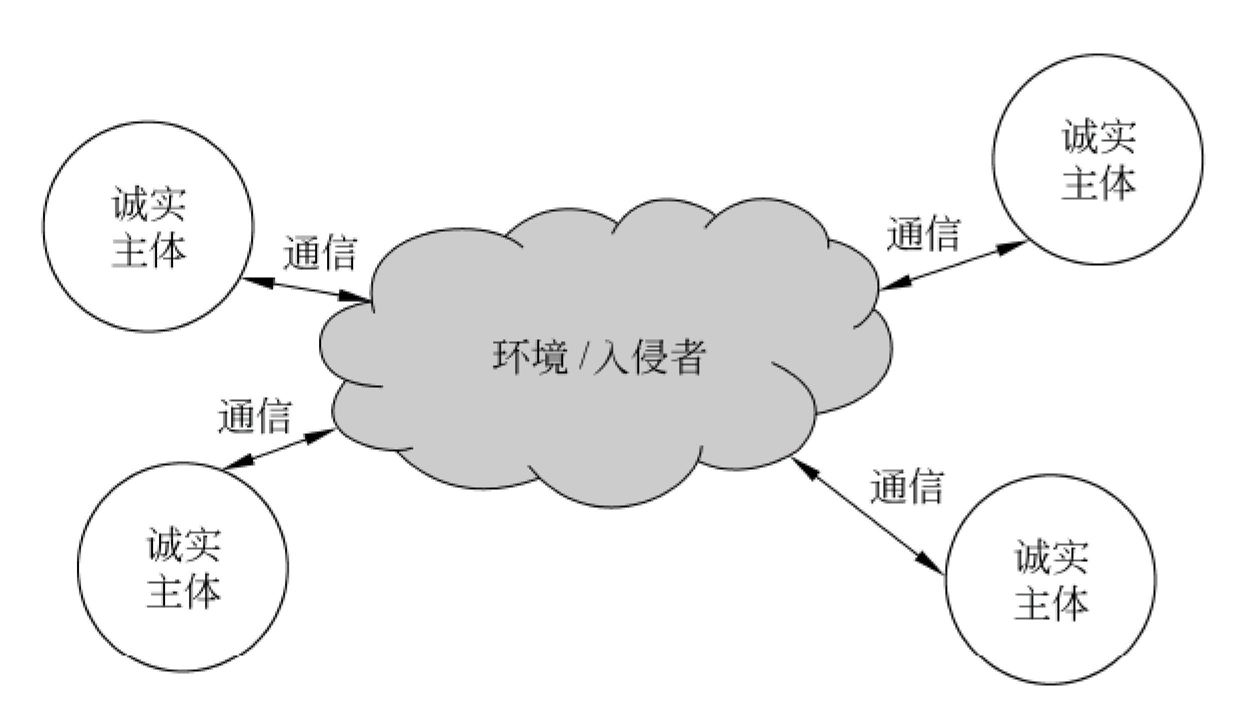


图 2.1 安全协议系统模型示意图

2.3 安全协议的安全属性

简单地说，安全协议的目标就是保证某些安全属性在协议执行完毕时能够得以实现，换言之，评估一个安全协议是否是安全的就是检查其所要达到的安全属性是否受到入侵者的破坏。主要有认证性、机密性、完整性和非否认性等安全属性。

(1) 认证性。认证可以对抗假冒攻击的危险，认证可以用来确保身份，并可用于获取对某人或某事的信任。在协议中，当某一成员(声称者)提交一个主体身份并声称它是那个主体时，需要运用认证以确认其身份是否如其声称所言。或者声称者需拿出证明其真实身份的证据，这个过程称为认证的过程。在协议的实体认证中可以是单向的也可以是双向的。

(2) 机密性。机密性的目的是保护协议消息不被泄露给非授权拥有此消息的人，即使是入侵者观测到了消息的格式，它也无法从中得到消息的内容或提炼出有用的信息。保证协议消息机密性的最直接的办法是对消息进行加密。加密使得消息由明文变为密文，并且任何人在不拥有密钥的情况下是不能解密消息的。

(3) 完整性。完整性的目的是保护协议消息不被非法改变、删除和替代。最常用的方法是封装和签名，即用加密或者签名的办法或者用 Hash 函数产生一个明文的摘要附在传送的消息上，作为验证消息完整性的依据，称为完整性校验值(ICV)。一个关键性的问题是，通信双方必须事先达成有关算法的选择等款项的共识。如果被保护的消息拥有一定的冗余，加密消息的冗余能保证消息完整性的效果。因为如果一个入侵者不知道加密密钥而修改了密文的一部分，则会导致在解密的过程中产生不正确的结果。

(4) 非否认性。非否认性的目的是通过通信主体提供对方参与协议交换的证据以保证其合法利益不受侵害，即协议主体必须对自己的合法行为负责，而不能也无法事后否认。非否认协议的主体目的是收集证据，以便事后能够向可信仲裁证明对方主体的确发送了或接收了消息。证据一般是以签名消息的形式出现的，从而将消息与消息的意定发送者进行了绑定。

另外，还有公平性、匿名性、可用性、可控性等安全属性，这里只对公平性和匿名性做一点解释，其他安全属性就不再赘述了。

(5) 公平性。公平性的目的是保证协议的参与者不能单方面中止协议或获得有别于其

他参与者的额外优势，在合同签署协议中具有重要的意义。

(6) 匿名性。匿名性的目的是保证消息的发送者的身份不被泄露，也就是消息与消息发送者的身份不再绑定在一起。

2.4 安全协议的设计准则

如果在安全协议的设计阶段就能够充分考虑一些不当的协议结构可能使协议的安全性免遭破坏，从而避免不必要的协议错误，将是事半功倍的。Abadi 和 Needham 在文献[1]中提出了设计安全协议应遵守的一些原则，归纳起来有以下几个方面。

(1) 消息独立完整性原则。每条消息都应能够准确地表达出它所想要表达的含义，一条消息的解释应完全由其内容来决定，而不必借助于上下文来推断。协议消息采用标准的形式化语言加以描述，或者采用非形式化的自然语言描述。此原则指出对接收消息的不同解释与不同的消息格式的假设有关。最常用的协议消息描述格式为：

$$[1]\ A \rightarrow B: m$$

协议设计者的本意是想表示：协议的消息[1]是由主体 A 向主体 B 发送 m。但是在具体的协议执行时这条消息会有多种可能的解释。例如，m 的发送者不一定保证是 A，m 的接收者也不能保证就是 B，或者 B 不是 m 的唯一接收者。原因在于 m 并未与主体进行正确的绑定，或者说消息内容并未完全反映设计者的意图，需要借助于上下文来分析。这就使入侵者可用此消息替换该协议其他轮次中的消息。

(2) 消息前提准确性原则。与消息的执行相关的先决前提条件应当明确给出，并且其正确性与合理性能够得到验证，由此可判断出此消息是否应当被接受。这条原则是在上一条原则的基础上的进一步说明，即不仅要考虑消息本身，还要考虑与每条消息相关的条件是否是合理的，或者说，每条消息所基于的假设是否是能够成立的。

(3) 主体身份标识原则。如果一个主体的标识对于某个消息的含义是重要的，那么最好在消息中明确地附上主体的名称。有两种方式：一种是显式的，即在消息中主体的名字以明文形式出现；另一种是隐式的，即采用加密或签名算法，使得能够从消息格式中明确地推知消息所属主体的身份。

(4) 加密目的明确性原则。明确采用密码运算的目的，否则将造成冗余。密码算法并不与安全性同义，它的不正确使用可导致协议错误。因此在使用密码算法时必须知道为什么使用以及如何使用。加密可实现多种安全目的，如机密性、完整性、认证性等，因此在协议中使用加密算法时，必须确保它的确能够保证某种安全属性的实现。

(5) 签名含义清晰性原则。当主体对一个加密消息进行签名时，并不表明主体知道加密消息的内容。反之，如果主体对一个消息签名后再加密，则表明主体知道消息的内容。因此，如果需要同时使用加密与签名时，应当先对消息进行签名再对结果进行加密。

(6) 临时值使用原则。在协议中使用临时值时，对其所具有的属性和所起的作用一定要认识清楚。如果使用一个可预测的值作为临时值，那么应该保护这个临时值以使入侵者不能模拟一个挑战而后重放响应。

(7) 随机数使用原则。在协议中使用随机数时，应明确其所起的作用和属性。使用随机数可提供消息的新鲜性，因此随机数的真正随机性是关键。

(8) 时间戳使用原则。当使用时间戳时，必须考虑各个机器的时钟与当地标准时间的差异，这种差异不能影响到协议执行的有效性。时间系统的维护成为可信任计算基础的重要部分。时间戳的使用极大地依赖于时钟的同步，但要做到这一点是很不容易的。

(9) 密钥使用原则。当使用密钥时，必须考虑密钥是否最近被使用过。例如，用于加密一个随机数，那么该密钥就是过期的或可能已被泄露。

(10) 消息定位原则。推测出一条消息属于哪个协议，属于该协议的哪次运行，并知道它在协议中的序号是可能的。

(11) 信任关系明确性原则。协议中的信任关系必须被明确地表达出来，并对这些信任关系的必要性给出合理的解释。

2.5 安全协议的缺陷分类

非正式地讲，如果一个安全协议使得非法用户不能从协议中获得比此协议本身所体现的更多的有用信息，则称这个协议是安全的，同时也意味着该协议能够达到预定的安全目标。

安全协议是许多分布式系统安全的基础，确保这些协议的安全运行是极为重要的。大多数安全协议只有为数不多的几个消息传递，其中每一个消息都是经过巧妙设计的，消息之间存在着复杂的相互作用与制约，而且在安全协议中往往使用多种不同的密码算法，此外协议设计者对协议运行环境的安全需求估计不足或者采用不当的技术，这些都会导致设计的安全协议存在安全漏洞与缺陷，并引发入侵者对协议进行各类攻击行为，从而破坏协议的安全性。

归纳起来，安全协议的缺陷从来源上讲可分为两类：一类是由于设计时的不规范引发的；另一类是在具体执行时产生的。学术界提出的安全协议设计原则，试图从一开始就避免协议缺陷产生的可能。

尽管协议设计者尽可能在协议设计时回避可能的人为错误，但是安全协议在实际应用时仍会出现各种类型的缺陷，并且产生的原因是十分复杂的，很难有一种通用的分类方法将安全协议的安全缺陷进行分类。Gritzalis 和 Spinellis 根据安全协议缺陷产生的原因和相应的攻击方法对安全缺陷进行了分类。

(1) 基本协议缺陷。它是指在安全协议的设计中没有或很少防范入侵者攻击而引发的协议缺陷。例如，对加密的消息进行签名，从而使入侵者通过用它自己的签名替换原有的签名来伪装成发送者。

(2) 口令/密钥猜测缺陷。这类缺陷产生的原因是用户往往从一些常用的词中选择其口令，从而导致入侵者能够进行口令猜测攻击；或者选取了不安全的伪随机数生成算法构造密钥，使入侵者能够恢复该密钥。

(3) 陈旧消息缺陷。它是指协议设计中对消息的新鲜性没有充分考虑，从而使入侵者能够进行消息重放攻击，包括消息源的攻击、消息目的的攻击等。根据消息的来源与去向，陈旧消息攻击可分为消息来源攻击与消息目的地攻击。

(4) 并行会话缺陷。协议对并行会话攻击缺乏防范，从而导致入侵者通过交换适当的协议消息能够获得所需要的重要消息。并行会话攻击可使入侵者通过交换一定的协议消息

获得重要的信息。协议中主体的角色可区分为单一角色和多重角色，在单一角色协议中主体与其角色之间有一一对应关系，而在多重角色协议中则是一对多的关系。据此，并行会话缺陷又可分为并行会话单一角色缺陷和并行会话多重角色缺陷。

（5）内部协议缺陷。协议的可达性存在问题，协议的参与者中至少有一方不能够完成所有必需的动作而导致缺陷。

（6）密码算法缺陷。协议中使用的密码算法导致协议不能够完全满足所要求的机密性、认证性等需求而产生的缺陷。

2.6 消息重放攻击及其对策

消息重放攻击主要是指入侵者利用其消息再生能力生成诚实用户所期望的消息格式并重放，从而达到破坏协议安全属性的目的。由于消息是协议的主要组成部分，协议主体的身份认证、消息的机密性都是通过消息的正确传递来实现的，因此根据攻击目的对消息进行任意的组合并重放是入侵者对协议实施各种具体攻击常用的手法。

Syverson[2]根据消息的来源和去向对消息重放攻击进行了分类，其中，根据消息的来源把重放攻击分为轮内攻击和轮外攻击。前者是指对一个协议轮内的消息进行重放，后者是指对一个协议的不同轮次的消息进行重放。对于协议的轮外攻击，根据协议的不同轮次的执行时间是否是重叠的，又可分为交叉攻击和典型重放攻击。前者不同轮次的协议执行时间是有重叠的，而后者则没有重叠。

根据消息的去向，可将重放攻击分为偏转攻击和直接攻击。偏转攻击是指改变了消息的去向，使消息为非意定的主体接收。又可区分为两种情况：一种是将消息返还给了发方，称为反射攻击；另一种是将消息发给了协议合法通信双方之外的任一方，称为第三方攻击。直接攻击是指消息的确发给了意定的接收方，但是被延迟了。

因此，当一个诚实的主体遭受重放攻击时，它收到的消息有可能是：本轮内的消息重放，无重叠轮外消息的重放，有重叠轮外消息的重放，延迟的消息。它发送的消息有可能是：被返还，被发往第三方，被延迟。

在消息重放攻击实施的基础上还可实现其他一些攻击。例如，G. Lowe[3]提出了一种攻击方法，多重会话攻击。多重会话主要是指通信双方建立了超出正常范围的会话数。具体而言，主体 A 欲与主体 B 建立一次会话，但由于入侵者的介入，却与主体 A 建立了两次甚至多次会话，并且 A 对此毫无觉察。多重会话的直接结果是对协议认证性的破坏，因为主体之间建立会话是在确认对方身份的基础上进行的。造成多重会话攻击的原因是通信双方都允许建立多重会话，并且对于入侵者的消息重放通信双方不能识别。再如窃听重放攻击，入侵者插入到声称者和验证者之间的通道上，主动修改使得它好像对验证者来说是一个合法的声称者，而对声称者来讲它是一个合法的验证者。

防止消息重放攻击的关键是保证消息的"新鲜性"。无论是基于对称密码算法还是公钥密码算法的安全机制，都要有一种机制保证交换消息是"新鲜"的。这里的"新鲜"是指消息刚刚被产生和发送，而不是先前产生和发送的消息的重放。通过保证消息的"新鲜"可有效地阻止入侵者实施各类消息的重放攻击。

实体认证在保证数据的机密性和认证性上是非常重要的。认证的目的就是保证通信双

方可以互相证实。通信实体是通过提交所知道的某项秘密信息来证实自己的身份。采用密码算法可以保证实体在此过程中不会将秘密信息泄露出去。但如果该实体总是使用同样的消息来证明身份，那任何人只需重放这个消息就可以伪装该实体，这也是入侵者实施消息重放攻击得手的原因。

因此，在认证协议中必须有一种机制能够检验消息的新旧以抵御重放。根据在消息中所放入的非重复值的不同，可分为 3 种方法：序列号机制、时间戳机制和挑战-响应机制，这 3 种机制都是基于时间变量的。

(1) 序列号机制。这一方法最为简单，接收方通过比较消息中的序列号以判断消息是新产生的还是重放的。一般而言，这一方法要求每对通信实体必须存储一对专用于消息"新鲜性"检测的序列号。而且这对通信实体必须事先商定好序列号递增的方式。实体收到消息后，根据事先协商的策略检查消息中的序列号是否有效。这样，由于每个消息都有其各自不同的序列号，且不会重复，接收方就可区分消息的新旧。

序列号机制要求每个消息都带有足够的信息以便接收方检验消息的新旧。这一特点使得安全通信前的延迟大大减少。这点对使用无连接服务的分布式应用尤为有用。但是以序列号作为认证协议中的变量参数，将会在分布式应用的管理上产生大量的开销。因为通信实体必须为与它通信的每个实体记录消息最后使用的序列号，即如果在一个分布式系统中存在 N 个实体，则每个实体必须存储($N-1$)对序列号。这在一个大环境中是不可行的。一般这种机制用于通信系统中成员较少的情形下。

(2) 时间戳机制。在这种机制中，消息的新旧是由消息上盖的时间戳决定的。只有当消息上的时间戳与当前本地时间的差值在一定范围内，接收方才接收这个消息。时间戳机制与序列号机制非常类似，通信双方只需交换两个消息就可互相认证。

但这一机制有一个致命的弱点：就是必须有一个全局时钟。而在实际的分布式系统中，各机器的时钟都是通过某种时钟同步协议来保证同步的。但目前不少时钟同步协议中缺少认证，而且许多用户没有意识到危险的存在，仍在继续使用它。即便采用安全级别较高的时钟同步协议，基于时钟的认证协议的安全强度也仍是值得怀疑的。因为消息的传递具有有限的速度，而各分布时钟不可能在任一时刻具有相同的时钟值。这就包含一些潜在的可能攻击：如果验证者弄错了当前的时间，那么旧消息就能被很容易地重放；如果一个合法的声称者弄错了当前的时间，那么就有可能被利用在一个合理的时间点对验证者重放后产生认证请求；如果双方的时钟都有较大的偏差，则双方都会被入侵者利用。

所以接收方必须考虑接收到的消息上的时间戳不可能同其本地时钟完全一致，为此采用"接收窗口"机制。其中窗口的尺寸必须足够大，以便能够接收大多数新的消息；而且也必须足够小，以便能够检查出入侵者的重放消息。然而，这一点在现实中往往很难达到，因为发送方无法保证所发送的新消息比入侵者发送的重放消息在网络上传递的时间短。所以无论接收窗口的尺寸有多小，它都可能比实际的最小重播延迟大。这一点使得基于时间戳的认证机制不能用于敏感性环境中。如目前广泛使用的 Kerberos V4 中就是利用时间戳抵御重播的，但它是建立在这样的一种假设上：在认证的生存周期里(如 5min 内)不存在重播现象。这一假设给该机制带来一个安全隐患。

(3) 挑战-响应机制。它也称口令-应答机制(Challenge-response)。在这一机制中，消息的时间变量参数由接收方在该消息传递前明确地向消息发送方说明。由于在该机制中采

用了多次握手以达到认证,因此系统的开销大大增加,至少在每次安全通信开始前,都需要一轮消息交换。这使得数据消息的认证不太可能,且该机制要求所有通信实体都必须保留状态信息、随机挑战值,以便完成认证协议。这一机制可方便地应用于基于 TCP 的服务中,但对那些基于 UDP 的服务来说,需要比较大的改动。因此,挑战-响应机制只适用于那些面向连接的通信中。

2.7 安全协议基础理论与方法概述

安全协议基础理论与方法的研究主要包括以下 5 个方面:可证明安全性理论与方法、形式化分析理论与方法、混合理论与方法、零知识证明理论与方法、安全多方计算理论与方法。本节简要概述这些基础理论与方法的研究目的、研究背景和发展过程等。

(1) 可证明安全性理论与方法。目前多数安全协议的设计现状是:

① 提出一种安全协议后,基于某种假想给出其安全性论断;如果该协议在很长时间(如 10 年)仍不能被破译,大家就广泛接受其安全性论断。

②一段时间后可能发现某些安全漏洞,于是对协议再作必要的改动,继续使用,这一过程可能周而复始。

这样的设计方法存在以下问题:

① 新的分析技术的提出时间是不确定的,在任何时候都有可能提出新的分析技术;

② 这种做法使人们很难确信协议的安全性,反反复复地修补更增加了人们对安全性的担心,也增大了实现代价或成本。可证明安全性理论与方法就是为解决上述问题而提出的一种解决方案(当然并非是唯一解决方案)。

可证明安全性是指,安全方案或协议的安全性可以被"证明",但用"证明"一词并不十分恰当,甚至有些误导。一般而言,可证明安全性是指这样一种"归约"方法:首先确定安全方案或协议的安全目标。例如,在加密方案中,其安全目标是确保信息的机密性;然后构造一个形式敌手模型,并且定义它对安全方案或协议的安全性"意味"着什么,对某个基于"极微本原"(Atomic Primitives,指安全方案或协议的最基本组成构件或模块,如基础密码算法、某数学难题等)的特定方案或协议,基于以上形式化模型去分析它,"归约"论断是基本工具;最后指出(如果能成功),挫败方案或协议的唯一方法就是破译或解决"极微本原"。换句话讲,对协议的直接分析是不必要的,因为对协议的任何分析结果都是对极微本原安全性的分析。从以上角度来看,称"归约安全"也许比"可证明安全"更恰当,但现在纠正为时已晚,无妨仍沿用原名称,但必须理解其真正含义。

综上所述,可证明安全性理论本质上是一种公理化研究方法,其最基础的假设或"公理"是:"好"的极微本原存在。安全协议设计难题一般分为两类:一类是极微本原不可靠造成协议不安全;另一类是即使极微本原可靠,安全协议本身也不安全,也就是其结构有缺陷。后一种情况更为普遍,是可证明安全性理论与方法的主要研究范围。

20 世纪 80 年代初,Goldwasser、Micali 和 Rivest 等人首先比较系统地阐述了可证明安全性这一思想,并给出了具有可证明安全性的加密和签名方案。不幸的是,以上方案的可证明安全性是以严重牺牲效率为代价的,因此以上方案虽然在理论上具有重要意义,但不实用,这种情况严重制约了这一领域的发展。直到 20 世纪 90 年代中期出现了"面向实际的可

证明安全性(Practice-Oriented Provable-Security)"概念,特别是 Bellare 和 Rogaway 提出了著名的 RO(Random Oracle,随机预言)模型方法论,才使得情况大为改观:过去仅作为纯粹理论研究的可证明安全性理论,迅速在实际应用领域取得重大进展,一大批快捷有效的安全方案相继提出;同时还产生了另一个重要概念:"具体安全性"(Concrete Security or Exact Security),其意义在于,人们不再仅仅满足于知道安全性的渐近度量,而是可以确切了解较准确的安全度量。面向实际的可证明安全性理论取得了巨大的成功,已为学术界和产业界广为接受;但 Canetti 和 Goldreich 对此持有异议,并坚持仍在标准模型(Standard Model)中考虑安全性。但可以肯定的是,迄今为止,RO 模型方法论是可证明安全性理论最成功的实际应用,其现状是:几乎所有国际安全标准体系都要求提供至少在 RO 模型中可证明的安全性设计,而当前可证明安全性的方案也大都基于 RO 模型。

(2) 形式化分析理论与方法。安全协议的形式化分析理论与方法可使协议设计者通过系统分析,将注意力集中于接口、系统环境的假设、系统在不同条件下的状态、条件不满足时系统出现的情况及系统不变的属性,并通过系统验证,提供协议必要的安全保证。通俗地讲,安全协议的形式化分析方法是采用一种正规的、标准的方法对协议进行分析,以检查协议是否满足其安全目标。因此,安全协议的形式化分析理论与方法的研究有助于:界定安全协议的边界,即协议系统与其运行环境的界面;更准确地描述安全协议的行为;更准确地定义安全协议的特性;证明安全协议满足其说明,以及证明安全协议在什么条件下不能满足其说明。

早在 1978 年,Needham 和 Schroeder 就提出了对安全协议进行形式化分析的思想,他们提出了为进行共享和公钥认证的认证服务器系统的实现建立安全协议,称之为 Needham-Schroeder 协议。1981 年 Denning 和 Sacco 指出了 Needham-Schroeder 协议的一个错误,使得人们开始关注安全协议形式化分析这一研究领域。真正在这一领域首先做出工作的是 Dolev 和 Yao。随后,Dolev、Even 和 Karp 等人也做了大量的工作。直到 1989 年,Burrows、Abadi 和 Needham 提出了 BAN 逻辑之后才打破了形式化分析这一领域的神秘感,并从此引起人们的关注。BAN 逻辑通过对认证协议的运行进行形式化分析,来研究认证双方通过相互发送和接收消息从最初的信仰逐渐发展到协议运行最终要达到的目的——认证双方的最终信仰。BAN 逻辑的规则十分简洁和直观,易于使用。BAN 逻辑成功地对 Needham-Schroeder 协议、Kerberos 协议等几个著名的协议进行了分析,找到了这些协议的已知的和未知的漏洞,并导致许多安全协议形式化分析方法的诞生。

安全协议的形式化分析方法目前主要有 3 类。第一类是以 BAN 类逻辑为代表的基于推理结构性方法,该方法主要是运用逻辑系统从用户接收和发送的消息出发,通过一系列的推理公理推证协议是否满足其安全说明。第二类是以基于 Dolev-Yao 模型的状态探测和定理推证和 Mur ϕ 模型检测为代表的基于攻击结构性方法,该方法对协议进行分析时一般要借助于自动化工具,如一般目的的模型检测工具 FDR 和 Mur ϕ 以及特殊目的的 NRL 分析器和 Interrogator,这些方法都是从协议的初始状态开始,对合法主体和一个攻击者的所有可能的执行路径进行穷尽搜索,以期找到协议可能存在的错误。第三类是以 Strand Space 理论和 Paulson 归纳法为代表的基于证明结构性方法,该方法的基本思路是推广和完善协议模型,根据新模型提出有效的分析理论。

(3) 混合理论与方法。安全协议的形式化方法与可证明安全性方法在相当长时间内独

立发展、自成体系。严格来讲，这两种方法是互斥的。可证明安全性方法具有明确的安全定义和计算化的语义，它所描述的安全强度、敌手模型都是比较接近于实际的，但实际可操作性较差，证明过程复杂，难以进行自动化分析；形式化方法在形式化分析与证明中利用的只是对安全协议形式系统的语构(Syntax)的研究，在语构和推理规则约定后，形式化系统的内部推演只是对一系列符号串的重写而已，这些过程都可以机械地完成，因此易于进行自动化分析，但形式化方法没有计算化的语义(Semantic)，语义不清楚，不可以量化安全，也不能像可证明安全性方法那样进行安全性归约证明。但是，近几年一些学者逐步认识到这两种方法具有很强的互补性：一方面，通过把形式化的语构推理引入到可证明安全性方法中，寻找密码可靠的形式化模型，使复杂的归约过程得到简化；另一方面，对形式化方法重新给出其计算化的语义，研究在语构下逻辑等价的两个变元在语义化后的计算等价性问题，即不可区分性问题，这就是所谓的形式化方法的计算合理性问题。Abadi 和 Rogaway 最早(2000 年)研究了将形式化的消息串计算化后的计算合理性问题。随后 Canetti 在 UC(Universal Composability)的基础上提出了符号化 UC 的概念，将符号化引入到可证明安全性方法中；Mitchell 领导的研究小组提出了计算化的符号验算模型；Backes 等提出了密码化(密码可靠)的形式化证明模型，这些工作都是对混合方法的具体尝试。目前混合理论与方法的研究主要有两大类：一类是形式化方法的计算合理性；另一类是密码可靠的形式化模型与方法。

(4) 零知识证明理论与方法。20 世纪 80 年代初，Goldwasser 等人提出了零知识证明这一概念。零知识证明是一种协议，这种协议的一方称为证明者，它试图使被称为验证者的另一方相信某个论断是正确的，却不向验证者提供任何有用的信息。Goldwasser 等人提出的零知识证明是交互式的，也就是证明者和验证者之间必须进行交互，才能实现零知识性，因而称为交互零知识证明。20 世纪 80 年代末，Blum 等人通过利用一个共同的称为参考串的短随机串代替交互实现了零知识证明这一思想，他们把这种零知识证明称为非交互零知识证明，这种证明是非交互的、单向的，也就是证明者和验证者在定理证明阶段无须进行交互，就能实现零知识性。非交互零知识证明比交互零知识证明的适用范围更广，因此大大地扩充了零知识证明思想的应用。

在交互零知识证明(Interactive Zero Knowledge Proofs)研究中，最常用的基本模型有两种。一种是 GMR 模型，在这种模型中，证明者具有无限的计算能力，验证者具有多项式时间的计算能力，证明指的是语言成员问题，即输入 I 是否是语言 L 的一个成员。GMR 的零知识证明不是真正的零知识证明，这是因为在证明中，证明者向验证者揭露了 1b 信息，即 $I \in L$。但除此之外，再没有其他任何附加的信息泄露给验证者，通常称这种交互零知识证明为成员或定理的零知识证明(Zero Knowledge Proofs of Membership or Theorem)。另一种是 FFS 模型，在这种模型中，证明者和验证者均具有多项式时间的计算能力，证明者的目的不是向验证者证明 $I \in L$，而是证明他知道 I 关于 L 的状况。FFS 的零知识证明是真正的零知识证明，因为在证明中，验证者没有得到任何信息，他连 $I \in L$ 还是 $I \notin L$ 都不知道，但他相信这个证明，通常称这种交互零知识证明为知识或身份的零知识证明(Zero Knowledge Proofs of Knowledge or Identity)。

在非交互零知识证明(Non-Interactive Zero-Knowledge Proof)研究中，人们也主要关心两种模型：一种是成员或定理的非交互零知识证明系统；另一种是知识的非交互零知识证明系统。

（5）安全多方计算理论与方法。安全多方计算的研究目标是为分布式计算提供安全解决方案。安全多方计算理论与方法的研究有助于：澄清分布式计算中一些最基本的安全问题；说明在既定的安全模型下哪些计算功能是可以安全实现的，哪些是不可行的；给出设计分布式安全协议的一般技术和方法；设计可应用于实际系统中的某些具体的方案和模块。

安全多方计算是由 Yao 于 1982 年提出的一个概念，Goldreich-Micali-Wigderson 给出了一般的描述。20 世纪 80 年代，安全多方计算的研究工作主要集中于安全模型的建立、安全概念的定义，以及安全计算功能的可实现性等基本问题。20 世纪 90 年代以来，主要研究有效的安全多方计算协议，并针对一些特殊问题，研究有效的非交互的解决方案，如联合解密、联合签名、保密数据库访问、保密信息检索（PIR）等。最近的研究工作主要集中于协议合成的安全性问题。

已有研究结果表明，理论上任何安全多方计算问题都可以通过电路计算协议来解决，但如何设计有效的安全多方计算协议，降低协议的交互通信轮数、通信的复杂性及计算复杂性，一直是人们关注的焦点。就目前来看，按照这一方法所设计的协议的效率与实用性之间的差距，可能是无法令人满意的。所以，对于特定问题，需要考虑其具体的安全环境，建立新的安全模型，重新定义可以接受的确切安全性，设计实用的安全多方计算协议。

2.8 小结

为了更好地理解安全协议的本质，本章简要介绍了安全协议的一些基本问题。安全协议的研究逐渐从密码学特别是密码算法的研究中分离出来（当然二者有着千丝万缕的联系），成为信息安全的一个独立的学科分支，其研究内容主要包括两大方面：安全协议的研究和协议的安全性研究。其最大特点是安全性分析难度大，与实际应用结合紧密。

安全协议理论方面的研究成果可参阅文献[4]～[7]，安全协议应用方面的研究成果可参阅文献[8]、[9]等。

参考文献

[1] Abadi M, Needham R. Prudent engineering practice for cryptographic protocols. IEEE Transactions on Software Engineering, 22(1): 6～15, January 1996.

[2] Paul Syverson. A Taxonomy of Relay Attacks. In proceedings of the 7th IEEE Computer Security Foundations Workshop, 97～101. ACM Press, New York, November, 131～136, 1994.

[3] Lowe G. A hierarchy of authentication specifications. In Proceedings of the 1997 IEEE Computer Society Symposium on Research in Security and Privacy, pages 31～43, 1997.

[4] 范红，冯登国. 安全协议理论与方法. 北京：科学出版社，2003.

[5] 冯登国. 可证明安全性理论与方法研究. 软件学报，Vol. 16，No. 10，2005，1743～1756.

[6] 冯登国，范红. 安全协议形式化分析理论与方法研究综述，中国科学院研究生院学报，Vo. 20，No. 4，2003，389～406.

[7] 薛锐，冯登国. 安全协议的形式化分析技术与方法，计算机学报，2006，29(01)，1～20.

[8] 冯登国. 计算机通信网络安全. 北京：清华大学出版社，2001.

[9] 陈性元，杨艳，任志宇. 网络安全通信协议. 北京：高等教育出版社，2008.

第2篇 安全协议基础理论与方法

安全协议基础理论与方法是构建安全协议的核心基础，是指导设计和分析安全协议的有力工具，其研究成果涉及面广、难度大、内容丰富。本篇重点介绍可证明安全性理论与方法、形式化分析理论与方法、混合理论与方法、零知识证明理论与方法、安全多方计算理论与方法等5个方面的内容，以便读者较全面地掌握和运用这些基础理论与方法，更好地理解安全协议设计和分析过程中涉及的一些基本概念、模型、方法和技巧。

第3章　可证明安全性理论与方法

启发式设计方法存在以下问题：新的分析技术的提出时间是不确定的，在任何时候都有可能提出新的分析技术；这种做法使人们很难确信协议的安全性，反反复复地修补更增加了人们对安全性的担心，也增大了实现代价或成本。

那么有什么解决办法呢？可证明安全性理论与方法就是为解决上述问题而提出的一种解决方案（当然并非是唯一解决方案）。

实际上，可证明安全性理论与方法是在一定的敌手模型下证明了安全方案或协议能够达到特定的安全目标，因此，合适的安全目标定义、适当的敌手模型是我们讨论可证明安全性的前提条件。

可证明安全性理论与方法的应用价值是显而易见的，一方面，可以把主要精力集中在“极微本原”的研究上，这是一种古老的、基础性的、带有艺术色彩的研究工作；另一方面，如果你相信极微本原的安全性，不必进一步分析协议即可相信其安全性。

必须说明的是，可证明安全性理论与方法也有局限性。首先必须注意模型规划，即注意所建模型都涵盖了哪些攻击，显然一些基于物理手段的攻击都不包含在内，但这并不意味着可证明安全性的方案就一定不能抵抗这类攻击，而是说未证明可以抵抗这类攻击；其次即使应用具有可证明安全性的方案，也可能有多种方式破坏安全性：有时证明了安全性，但问题可能是错误的，也可能应用了错误的模型或者协议被错误操作，甚至软件本身可能有错误。

另一需要注意的问题是基础假设的选取。可证明安全性是以某一假设为基础的，因此一旦该假设靠不住，安全性证明也就没有意义（当然不一定意味着可构造对方案的攻击实例）；选取基础假设的原则就是“越弱越好”，通常称弱假设为标准假设。基础假设的强弱是比较不同安全方案的重要尺度之一。

上述表述较为抽象，下面以RSA为例加以说明。

给定某个基于RSA的协议 P，如果设计者或分析者给出了从RSA单向函数到 P 安全性的归约，那么 P 具有以下转换性质：对于任何声称破译 P 的敌手（程序）A，以 A 为“转换算法”的输入，必然导致一个协议 Q，Q 可被证明破译RSA。结论是：只要你不相信RSA是可破译的，那么上述的 Q 就不存在，因而 P 是安全的。

3.1　基本概念与计算假设

3.1.1　基本概念

对可证明安全性的精确形式化有多种形式，一般是在计算复杂性理论框架下讨论，如主要考虑“概率多项式时间（PPT）”敌手 A 和转换算法，以及“可忽略”的成功概率。这是一种“渐近”观点，有着广泛的适用范围。详细内容可参阅文献[1]。概率多项式时间算法实质上

是指有随机输入、并在其输入长度的多项式时间内一定有输出结果的算法。下面先给出可忽略函数的精确描述。

定义 3.1(可忽略函数) 设 $\mu(n):N\rightarrow R$,称函数 $\mu(n)$ 是可忽略的,如果对任意多项式 $p(\cdot)$,对于足够大的 n,满足 $\mu(n)<1/p(n)$。

定义 3.2(计算不可区分) 设 $\{X_n\}$ 和 $\{Y_n\}$ 是两个概率空间,称它们是(多项式时间)计算不可区分的,表示为 $\{X_n\}\overset{c}{\approx}\{Y_n\}$ 或 $\{X_n\}\approx\{Y_n\}$,如果对任意多项式 $p(\cdot)$,任意 PPT 算法 D 及所有辅助输入 $z\in\{0,1\}^{poly(n)}$,满足

$$|\Pr[D(X_n,1^n,z)=1]-\Pr[D(Y_n,1^n,z)=1]|<1/p(n)$$

定义 3.2 是说,不存在明显可区分概率空间 $\{X_n\}$ 和 $\{Y_n\}$ 的 PPT 算法,即在多项式时间内,任何 PPT 区分算法的成功概率总是可忽略的。

本质上,可证明安全性理论与方法的主要研究途径是规划安全方案或协议的形式化安全模型,不同的安全方案或协议会导致不同的安全模型,而这些安全模型大多都基于一些很基本的密码学概念。因此,对一些最基本的密码学概念(如加密、签名及其安全性定义等)给予精确的形式化定义,是可证明安全性理论与方法的基础组成部分,有助于消除自然语言的语义二义性。

定义 3.3(数字签名方案) 一个数字签名方案由以下 3 个算法组成。

(1) 密钥生成算法 K。对于输入 1^k,K 产生一对匹配值(k_p,k_s),分别称为公钥和私钥,K 可以是概率算法。k 称为安全参数,密钥等因素的规模都依赖于 k。

(2) 签名算法 Σ。给定消息 m 和(k_p,k_s),Σ 产生签名 σ,Σ 可以是概率算法。

(3) 验证算法 V。给定签名 σ、消息 m 和公钥 k_p,V 测试 σ 是否是 m 的对应公钥 k_p 的合法签名,通常情况下 V 是确定性算法,输出 1(表示签名有效)或 0(表示签名无效)。

对于任一数字签名方案(K,Σ,V),敌手 A 的模型如下。

A 的目标有以下 3 个:揭示签名者私钥(完全破译);构造成功率高的伪签名算法(通用伪造);提供一个新的消息-签名对(存在性伪造)。

存在性伪造一般并不危及安全,因为输出消息很可能无意义,但这样的方案本身不能确保签名方的身份,如不能用来确认伪随机元素(如密钥),也不能用来支持非否认。

A 的两类攻击:未知消息攻击和已知消息攻击。后一种情况中最强的攻击是"适应性选择消息攻击",即 A 可以向签名方询问对任何消息的签名(当然不能询问欲伪造消息的签名,这是一类自明的约定,后不注),因而可能根据以前的答复适应性地修改随后的询问。

定义 3.4(数字签名方案的抗适应性选择消息攻击安全性) 对任一数字签名方案(K,Σ,V),如果敌手 A 的攻击成功概率

$$\mathrm{Succ}_A=\Pr[(k_p,k_s)\leftarrow K(1^k),(m,\sigma)\leftarrow A^{\Sigma_{k_s}}(k_p):V(k_p,m,\sigma)=1]$$

是可忽略的,则称该方案能够抵抗适应性选择消息攻击。这里 A 可以获得签名 OracleΣ_{k_s}(实际上是一个"黑盒"),这模拟了以上所说的"适应性选择消息询问"。

定义 3.5(公钥加密方案) 一个公钥加密方案由以下 3 个算法组成。

(1) 密钥生成算法 K。对于输入 1^k,K 产生一对匹配值(k_p,k_s),分别称为公钥和私

钥，K 可以是概率算法。

(2) 加密算法 E。给定消息 m 和公钥 k_p，E 产生 m 对应的密文 c。E 可以是概率算法，这时记为 $E(k_p, m; r)$，r 表示随机输入。

(3) 解密算法 D。给定密文 c 和私钥 k_s，D 产生 c 对应的明文 m，D 一般是确定性算法。

一般而言，公钥加密方案的安全目标是单向性(One Wayness，OW)，即在不知道私钥的情况下，敌手 A 在概率空间 $M\times\Omega$ 上成功地对 E 求逆的概率是可忽略的(这里 M 是消息空间，Ω 是公钥加密方案的随机掷硬币空间)，亦即概率

$$\mathrm{Succ}_A = \Pr[(k_p, k_s) \leftarrow K(1^k): A(k_p, E(k_p, m; r)) = m]$$

是可忽略的。

然而，许多应用要求更强的安全性。

定义 3.6(多项式安全/密文不可区分) 对任一公钥加密方案(K, E, D)，如果满足

$$\begin{aligned}\mathrm{Adv}_A = 2\times\Pr[&(k_p, k_s) \leftarrow K(1^k), (m_0, m_1, s) \leftarrow A_1(k_p), c\\ &= E(k_p, m_b; r): A_2(m_0, m_1, s, c) = b] - 1\end{aligned}$$

是可忽略的，则称该方案是多项式安全的或密文不可区分的，这里敌手 $A=(A_1, A_2)$ 是一个 2 阶段攻击者(都是 PPT 算法)，概率取于(b, r)之上。

定义 3.6 形式化了以下性质：敌手了解明文某些信息(可任选一对消息，其中一个被加密)，但它不能从密文得到除明文长度之外的任何信息。

另一个更新的安全概念是“非延伸性”(Non-Malleability)，即敌手得到一个密文，敌手不能或以可忽略概率生成一个新的密文使得两个明文意义相关(如 DES 的互补明文结构)。该概念强于抗选择明文攻击的多项式安全性，但和抗选择密文攻击的多项式安全性是等价的。

敌手的几种攻击类型(相当于敌手拥有的 Oracle 数量及性质)如下。

(1) CPA(选择明文攻击)，该攻击在公钥加密方案中显然是平凡的。

(2) PCA(明文校验攻击)，敌手获得明文校验 Oracle，用以回答关于任一输入对(m, c)是否是对应明密文对的询问。

(3) CCA(选择密文攻击)，除获得加密 Oracle 外，敌手还获得解密 Oracle，即对于任何询问的密文(除了应答密文)，Oracle 都给予相应的明文作为回答。这是最强的攻击(根据是否适应性选择密文，还可以细分为 CCA1 和 CCA2)。

对应以上攻击条件的相应安全性定义，均可用类似于定义 3.6 的方法给出，区别仅在于敌手获得的 Oracle 数量和性质不一样。对称密码方案的安全性可类似定义。

3.1.2 计算假设

许多安全概念并不能在无条件的情况下得到保证，因此，安全性一般依赖于以下计算假设：单向函数的存在性、或者单向置换的存在性、或者是陷门单向函数(置换)的存在性。单向函数是一个满足以下条件的函数 f：任何人容易计算函数值，但是给定 $y=f(x)$，恢复 x(或 y 的任何原像)在计算上是不可行的。单向置换是一个双射的单向函数。对于加密处理来说，希望只有接收者才可以求逆，于是陷门单向置换是一个特殊的单向置换，其秘密信

息(即陷门)有助于对函数进行求逆。

给定计算假设“在没有陷门信息的情况下,计算函数的逆是不可行的”,希望不需要额外的假设即可得到安全性。形式上证明这一事实的唯一方法是证明“攻击安全方案或协议的敌手可以用于构造一个算法,该算法能够求解基础计算假设”。

在计算假设之间存在一个偏序关系:如果问题 P 比问题 P' 更困难(P' 归约到 P),那么问题 P 的困难性假设就比问题 P' 的困难性假设弱。所需要的假设越弱,安全方案或协议的安全性就越高。

目前主要使用以下两类计算假设。

(1) 整数分解与 RSA 问题。

(2) 离散对数与 Diffie-Hellman 问题。第一个使用的群是 $\mathbb{Z}_p^*$ 的循环子群,现在也常常使用基于椭圆曲线的循环子群。

1. 整数分解与 RSA 问题

最著名的困难问题是整数分解问题,即把两个素数 p 和 q 相乘得到 $n=p\cdot q$ 是容易的,而把合数 n 分解为素因子 p 和 q 则不是一件容易的事情。目前,最有效的整数分解算法是数域筛法(NFS)。数域筛法是超多项式的、亚指数算法,其复杂度是

$$\mathcal{O}(\exp((1.923+o(1))(\ln n)^{1/3}(\ln \ln n)^{2/3}))$$

该方法在 1999 年 8 月创造了一个记录,它把一个 155 位的整数(512b)分解为两个 78 位的素数之积。所分解的数称为 RSA-155,来自于“RSA 挑战列表”,用于衡量 RSA 密码体制的安全性,该体制在 SSL 握手协议的软件和硬件实现中广泛使用。

整数乘法只是提供了一个单向函数,没有任何可能来对其求逆。现在不知道如何使得整数分解更容易一些。但是,有一些代数结构是基于整数 n 的分解,其中某些计算在不知道 n 的分解的情况下是困难的,在知道 n 的分解的情况下很容易。如对有限环 $\mathbb{Z}_n$,如果 $n=p\cdot q$,则同构于 $\mathbb{Z}_p\times\mathbb{Z}_q$。

例如,对任何元素 x,计算其 e 次幂是容易的。但是,要计算 e 次根,看起来需要知道满足 $ed\equiv 1 \bmod \varphi(n)$ 的整数 d。这里 $\varphi(n)$ 是 Euler 函数,对于 $n=p\cdot q$ 这种特殊情形,$\varphi(n)=(p-1)(q-1)$。因此,$ed-1$ 是 $\varphi(n)$ 的倍数,故等价于 n 的分解。

1978 年,Rivest、Shamir 和 Adleman 定义了著名的 RSA 问题,并设计了首个公钥密码体制——RSA。

RSA 问题:设 $n=p\cdot q$ 是两个相同规模的大素数的乘积,e 是与 $\varphi(n)$ 互素的整数。

对给定的 $y\in\mathbb{Z}_n^*$,计算 y 的模 e 次根 x,即满足 $x^e=y \bmod n$ 的 $x\in\mathbb{Z}_n^*$。

设 $n=\prod p_i^{v_i}$,则 Euler 函数容易用下式计算,即

$$\varphi(n)=n\times\prod\left(1-\frac{1}{p_i}\right)$$

因此,利用 n 的分解(陷门),RSA 问题很容易求解。但是没有人知道是否必须利用 n 的分解来求解 RSA 问题,更不知道如何在不知道 n 的分解的情况下来求解 RSA 问题。

RSA 假设:对任何两个足够大的素数的乘积 $n=p\cdot q$,RSA 问题是难解的(可能与分解

n 一样困难)。

2. 离散对数与 Diffie-Hellman 问题

设($\mathbb{G}$,·)表示一个阶为 q 的循环群,其中 q 是素数。令 g 表示$\mathbb{G}$ 的一个生成元,即$\mathbb{G}=\langle g\rangle$。与离散对数相关的困难问题,以简洁的形式描述如下:

离散对数(DL)问题:给定 $y\in\mathbb{G}$,计算 $x\in\mathbb{Z}_q^*$,使得 $y=g^x$,记为 $x=\log_g y$。

计算 Diffie-Hellman(CDH)问题:对于任意的整数 $a,b\in\mathbb{Z}_q^*$,给定$\langle g,g^a,g^b\rangle$,计算 g^{ab}。

判定性 Diffie-Hellman(DDH)问题:对于任意的整数 $a,b,c\in\mathbb{Z}_q^*$,给定$\langle g,g^a,g^b,g^c\rangle$,判定是否有 $c\equiv ab \bmod q$ 成立。

上述问题显然是以从强到弱的顺序进行排列的,即 DL≥CDH≥DDH,其中 $A\geq B$ 表示问题 A 至少与问题 B 一样困难。然而,在实际中,没有人知道如何求解其中的任何一个问题,除非可以破解 DL 问题本身。而且,这些问题都是随机自归约的,即任何实例都可以归约到一个均匀分布的实例。例如,对于一个给定的元素 y,想要计算它对于基底 g 的离散对数 x。可以选择一个随机的 $t\in\mathbb{Z}_q$,计算 $z=ty$。那么 z 就是群中均匀分布的元素,而根据离散对数 $\alpha=\log_g z$ 就可以计算出 $x=\alpha/t$。因此,它们只有平均复杂情形,如果能够在多项式时间内求解不可忽略的部分实例,那么就可以在期望多项式时间内求解任何实例。

目前,求解离散对数问题最有效的算法依赖于其基础群。对于一般性的群(没有特定的代数性质可以应用),算法复杂度是 q 的平方根。但是,对于$\mathbb{Z}_p^*$ 的子群,存在一个更好的方法。最好的算法是基于数域上的筛法,正如因子分解问题一样。一般数域筛法的复杂度是超多项式且亚指数的:

$$\mathcal{O}(\exp((1.923+o(1))(\ln p)^{1/3}(\ln\ln p)^{2/3}))$$

在 2001 年 4 月,该算法创造了一个最新纪录,它对 120 位的素数 p 计算出了$\mathbb{Z}_p^*$ 中的离散对数。因此,只要一般攻击不能应用,而且其生成元的阶至少为 160b,512b 的素数仍然是足够安全的。

对于签名方案只要求群的离散对数问题困难即可,而对于加密方案需要陷门,因而要求群中的某些 DH 问题也是难以求解的。

3.2 随机预言模型方法论

20 世纪 80 年代初,Goldwasser、Micali 和 Rivest 等人首先比较系统地阐述了可证明安全性这一思想,并给出了具有可证明安全性的加密和签名方案[2,3]。但不幸的是,这些方案的可证明安全性是以严重牺牲效率为代价的,因此这些方案虽然在理论上具有重要意义,却完全不实用,这种情况严重制约了这一方向的发展。直到 20 世纪 90 年代中期出现了"面向实际的可证明安全性(Practice-Oriented Provable-Security)"概念,特别是 Bellare 和 Rogaway 提出了著名的随机预言(Random Oracle,RO)模型方法论[4],才使得情况大为改观:过去仅作为纯粹理论研究的可证明安全性理论与方法,迅速在实际应用领域取得重大

进展，一大批快捷、有效的安全方案相继提出；同时还产生了另一个重要概念“具体安全性”(Concrete Security or Exact Security)，其意义在于，人们不再仅仅满足于知道安全性的渐近度量，而是可以确切地了解较准确的安全度量。面向实际的可证明安全性理论与方法取得了巨大的成功，已为学术界广泛接受；但 Canetti 和 Goldreich 等对此持有异议[5]，并坚持仍在标准模型(Standard Model)中考虑安全性。

Canetti 和 Goldreich 认为，密码方案在 RO 模型中的安全性和通过“Hash 函数实现”的安全性之间无必然的因果关系。具体说来，存在这样的实际签名方案和加密方案，它们在 RO 模型中是安全的，但任何具体实现都是不安全的。这实际上是提出了一个反例。不过 Goldreich 也认为，应该明确 RO 模型方法论并不能作为实际方案安全的绝对证据，但该方法论仍是有意义的，如可以作为一种基本测试，任何实际方案通过这种安全测试是必要的，RO 模型方法论至少可以排除很多不安全设计，虽然并非完备的(即有些不安全设计可能排除不了)。Canetti 则进一步指出，RO 模型方法论虽然有以上缺点，但它可用于设计简单有效的协议，可以抵抗许多未知攻击；更重要的是，其基本思想可以用来设计某些安全的理想系统。

Pointcheval 等人则认为[6]，目前还没有人能提出令人信服的关于 RO 模型实际合法性的反例。文献[5]中的反例仅仅是一种理论上的反例，是针对实际目的的“明显错误设计”；RO 模型已经被广泛接受，并被认为是度量实际安全级别的一种很好的手段；即使并未提供一个正规的安全性证明(像标准模型那样)，但在其“安全性论断”(Hash 函数没有弱点)下，RO 模型中的证明确保了整个方案的安全性。更正规些，RO 模型可视为对敌手能力的某种限制，敌手的攻击是不考虑任何特殊 Hash 函数实例的一般攻击，而且如果假定存在某些防串扰设备(如 Smart Cards)，则 RO 模型等价于标准模型，这时只要求伪随机函数存在[1]。最重要的是，仅就实现效率这一点，RO 模型中的可证明安全性方案就远远优于那些提供标准安全性证明的安全方案，即仅此一点就可以从实际应用中排除当前所有“在标准模型中具有可证明安全性”的方案。事实上一些有代表性的有效标准解决方案，如文献[2]、[3]中的方案，过于复杂且代价昂贵，归约的复杂性使得难以确定实际安全参数，其有效性也只是相对过去的标准方案而言。

但可以肯定的是，迄今为止，RO 模型方法论是可证明安全性理论与方法最成功的实际应用，其现状是：几乎所有国际安全标准体系都要求提供至少在 RO 模型中可证明的安全性设计，而当前可证明安全性的方案也大都基于 RO 模型。

3.2.1 RO 模型介绍

文献[4]中提出以下观点：假定各方共同拥有一个公开的 RO，就在密码理论和应用之间架起了一座“桥梁”。具体办法是，设计一个协议 P 时，首先在 RO 模型(可看成一个理想模拟环境)中证明 P^R 的正确性，然后在实际方案中用“适当选择”的函数 h 取代该 Oracle (潜在论断是理想模拟环境和现实环境在敌手看来是多项式时间计算不可区分的)。一般来说，这样设计出来的协议可以和当前协议的实现效率相当。

必须指出，这并非是严格意义上的可证明安全性，因为安全性证明仅在 RO 模型中成

立，随后的“取代”过程本质上是一种推测：RO模型中的安全特性可以在标准模型中得以保持。

假设提出一个协议问题Π(这个问题和h函数“独立”)，要设计一个安全协议P解决该问题，可按以下步骤执行。

(1) 建立Π在RO模型中的形式定义，RO模型中各方(包括敌手)共享随机Oracle R。

(2) 在RO模型中设计一个解决问题Π的有效协议P。

(3) 证明P满足Π的定义。

(4) 实际应用中用函数h取代R。

严格来讲，h不可能真的像随机函数：首先其描述较短；其次，所谓的随机Oracle即Hash函数对每一个新的询问产生一随机值作为回答(如果问相同的询问2次，回答仍相同)，这也是和随机函数的一个微小区别。但这并未改变上述方法论的成功，因为只要求在敌手看来像随机函数。此外，h函数“独立”于Π也是至关重要的(否则可能不安全，可构造反例)。

一般来说，函数h至少要满足以下基本要求：设计上足够保守，能够抵抗各种已知攻击；不会暴露某些相关数学“结构”。文献[4]指出，选择h并不需要太麻烦，一个适当选择(但并不需过分苛求)的Hash函数就是以上h函数的一个很好选择。尽管SHA-1本身不是一个好的选择，但只需截短其输出或用某种非标准方式使用，如$h(x)=\text{SHA-1}(xx)$。

RO方法论也易于推广到基于对称密码本原的协议/方案研究，如CBC-MAC，虽然没有Hash函数，但把一个恰当选择的分组密码(如AES)视为随机函数。

3.2.2 归约论断和具体安全性

归约论断是可证明安全性理论的最基本工具或推理方法，简单地说就是把一个复杂的协议安全性问题归结为某一个或几个难题(如大整数分解或离散对数等)。在RO模型中的归约论断一般表现为：首先形式化定义方案的安全性，假设PPT敌手能够以不可忽略概率破坏协议安全性(如伪造签名)；然后模仿者S(就是设计者或分析者)为敌手提供一个与实际环境不可区分的模拟环境(RO模型)，回答敌手的所有Oracle询问(模拟敌手能得到的所有攻击条件)；最后利用敌手的攻击结果(如一个存在性伪造签名)设法解决基础难题。如果把RO模型换成现实模型，就得到标准安全性证明。

RO归约论断的一个显著优点是能够提供具体安全性结果。具体地说，就是试图显式地得到安全性的数量特征，这一过程称为“具体安全性处理”(Concrete or Exact Treatment of Security)，与前面提到的“渐近”观点有明显区别。其处理结果一般表述为以下形式(举例)：“如果DES(本原)可以抵抗这样条件的攻击，即敌手至多获得2^{36}个明密文对，那么该协议可以抵抗一个能执行t步操作的敌手发动的攻击，t值如下……。”这样，协议设计者就能够确切地知道具体获得了多少安全保证，不必再笼统地说协议是否安全。

例3.1 文献[8]中研究了CBC-MAC的安全特征，结论是：对任意一个运行时间至多为t、至多见过q个正确MAC值的敌手，成功模仿一个新消息的MAC值的概率至多为$\varepsilon+(3q^2n^2+l)/2^l$，这里$l$是基础密码的分组长度，$n$是明文消息总数，$\varepsilon$是检测到密码偏离随机

行为的概率(在$\mathcal{O}(nql)$时间内)。

具体安全性处理的一个重要目标就是,在把一个基础极微本原转化成相应协议时,尽可能多地保持极微本原的强度。这表现为要求“紧”的归约方法,因为一个“松”的归约意味着要求采用更长的安全参数,从而降低了效率。

3.2.3 RO模型下安全的公钥加密和数字签名方案

1. 公钥加密方案

3.1节中的公钥加密方案的概念可直接推广到RO模型中,从而得到RO模型中的公钥加密方案的定义。公钥加密方案可通过PPT生成器g规定:以安全参数1^k为输入,输出一对概率算法(E,D),分别称为加密算法和解密算法,D保密,运行时间以g的运行时间为界。加密过程为:$y \leftarrow E^R(x)$;解密过程为:$x \leftarrow D^R(y)$。

像定义3.6一样,称g在RO模型中是CPA多项式安全的,如果对任意的选择明文敌手(F,A_1),满足

$$\Pr[R \leftarrow 2^{\infty};(E,D) \leftarrow g(1^k);(m_0,m_1) \leftarrow F^R(E);b \leftarrow \{0,1\}$$
$$\alpha \leftarrow E^R(m_b):A_1^R(E,m_0,m_1,\alpha) = b] \leqslant 1/2 + \mu(n)$$

这里,R表示一般的Oracle,是从$\{0,1\}^*$到$\{0,1\}^{\infty}$的函数,2^{∞}表示所有Oracle的集合,“∞”并非真的无限,只是避免提问“足够长是多长”这类问题,$\mu(n)$是可忽略函数。

CCA安全性:这里的敌手A称为RS敌手,即有非一致多项式算法$A=(F,A_1)$,各自获得一个Oracle R及一个解密Oracle的黑盒实现D^R;F的任务就是提出一对明文m_0,m_1,A_1被随机给予其中一个的密文α,则只要不允许向解密Oracle D^R询问α(因为禁止提出和最终论断等价的询问),A_1就不可能以不可忽略优势猜中是哪一个明文。

称g在RO模型中抗CCA攻击是安全的,如果对任意RS敌手(F,A_1),满足

$$\Pr[R \leftarrow 2^{\infty};(E,D) \leftarrow g(1^k);(m_0,m_1) \leftarrow F^{R,D^R}(E);b \leftarrow \{0,1\}$$
$$\alpha \leftarrow E^R(m_b):A_1^{R,D^R}(E,m_0,m_1,\alpha) = b] \leqslant 1/2 + \mu(n)$$

Bellare等在文献[4]中提出了一个在RO模型中抗CCA攻击是安全的方案,由于归约并不“紧”,应用意义并不大,但其设计思想很能体现RO方法论的特点。Bellare等把该思想作了进一步改进,在1994年提出了著名的公钥加密方案OAEP[9],可证明该方案抗CCA2攻击是安全的,目前已成为新一代RSA加密标准。基本组成如下:核心组件是一个Padding函数,即$\mathrm{OAEP}^{G,H}(x,r)=x \oplus G(r) \| r \oplus H(x \oplus G(r))$,这里$x$是被加密消息,$r$是随机输入;加密算法为$E^{G,H}(x)=f(\mathrm{OAEP}^{G,H}(x,r))$,这里$f$是陷门置换(如RSA函数)。基本设想是构造一个具有良好随机性的“遮掩函数”隐蔽明文的统计特性。

2. 数字签名方案

Bellare等在文献[4]中也给出了一种具有可证明安全性的签名方案,该签名方案要求陷门置换f具有“均匀分布”特点,而标准RSA置换不具有这个性质,因此基于RSA无法设计该类方案。文献[10]中提出了一个基于RSA的签名方案,该签名方案引入了概率机制,有更好的安全界。该方案不仅可证明其安全性,而且相应的归约是很“紧”的,一个敌手伪造

签名的能力和对 RSA 求逆的能力相当,总之安全性和分解整数的困难性紧密相关。稍作改进,也可以具有消息恢复功能。

目前其他可证明安全性的签名方案大都是基于识别协议的签名方案。例如,Fiat 和 Shamir 曾应用 RO 假设构造了一个安全性和分解因子一样困难的签名方案[7],并证明了其与识别协议的等价性。文献[11]对基于 Fiat-Shamir 识别协议的签名方案[7]作了具体安全性分析,通过交换应答和承诺的顺序改进设计了一类新的 Fiat-Shamir 类型签名方案(E-swap 签名方案),具有更好的具体安全性。

其他一些进展可参见文献[12]~[14]等。文献[14]基于计算 Diffie-Hellman(CDH)假设,对一个源于 Schnorr 签名的改进方案 EDL 给出了归约很"紧"的安全性证明,使得该签名方案得到了业界的广泛重视。另外值得特别说明的是,文献[12]对于完善 RO 模型方法论具有重大贡献,即提出了以 Folklore 引理为代表的一般性安全论断,主要适用于许多基于识别协议的签名方案,特别是证明了迄今为止唯一一个 ElGamal 变形签名方案 MEG 的安全性。Folklore 引理的基本思想是 Oracle 重放(Replay)攻击,即在 RO 模型中实施归约化证明时,重放多项式不同(但有一定联系)的随机 Oracles(这相当于为敌手提供多个模拟环境),如果敌手能以不可忽略的概率伪造多个签名,就可以求解该方案的基础困难问题,如离散对数问题。该方法的缺点是所得到的归约不够"紧"。

这里主要介绍 Bellare 和 Rogaway 于 1996 年提出的 RSA-FDH 签名方案[10],该方案的基本思想是结合 RSA 假设并基于经典的全域杂凑(Full-Domain Hash)签名方法论。该方案由以下 3 个算法组成。

(1) 密钥生成算法:输入 1^k,该算法随机选择两个 $k/2$b 的素数 p 和 q,计算 $n=p\cdot q$;随机选择 $e\in \mathbb{Z}_{\varphi(n)}^*$ 并计算 d 使得 $ed\equiv 1 \bmod \varphi(n)$。

用户的公钥为 (e,n),私钥为 (d,n)。设 $H:\{0,1\}^*\rightarrow \mathbb{Z}_n^*$ 是一个抗碰撞的 Hash 函数。

(2) 签名算法:给定消息 m 和私钥 (d,n),该算法计算并输出签名 $\sigma=H(m)^d \bmod n$。

(3) 验证算法:给定签名 σ、消息 m 和公钥 (e,n),该算法检验 $\sigma^e\equiv H(m) \bmod n$,如果成立,输出 1(签名有效),否则输出 0(签名无效)。

为了证明 RSA-FDH 的安全性,下面给出 RSA 问题的定量描述。

对于所有的 $k\in\mathbb{N}$,如果一个求逆算法 $\mathcal{I}$ 可以在 $t(k)$ 时间内以 $\varepsilon(k)$ 的成功概率破解 RSA 问题,则称算法 $\mathcal{I}$ 可 (t,ε) 破解 RSA 问题。

如果任何求逆算法都不能 (t,ε) 破解 RSA 问题,则称 RSA 是 (t,ε) 安全的。

定理 3.1 假设 RSA 问题是 (t',ε') 安全的,则 RSA-FDH 签名方案是 (t,ε) 安全的,其中

$$t = t' - (q_h + q_{\text{sig}} + 1)\cdot \mathcal{O}(k^3)$$

$$\varepsilon = \frac{1}{\left(1-\dfrac{1}{q_{\text{sig}}+1}\right)^{q_{\text{sig}}+1}}\cdot q_{\text{sig}}\cdot \varepsilon'$$

式中,q_{sig} 表示签名询问的次数;q_h 表示随机预言询问的次数。当 q_{sig} 较大时,有

$$\varepsilon \approx \exp(1)\cdot q_{\text{sig}}\cdot \varepsilon'$$

证明 假设伪造者$\mathcal{F}$经过 q_{sig}次签名询问和 q_h 次随机预言询问，可以(t,ε)攻破 RSA-FDH 签名方案。下面构造一个求逆算法$\mathcal{I}$可以(t',ε')破解 RSA 困难问题。

假设给定了实例 y，挑战者要找到满足 $x^e \equiv y \bmod n$ 的 x。注意到，在随机预言模型方法论中，伪造者$\mathcal{F}$不能自己计算 Hash 值，他只能通过 Oracle 询问来得到消息的 Hash 值。

算法$\mathcal{I}$设置用户的公钥为(n,e)，以此来运行算法$\mathcal{F}$。攻击者$\mathcal{F}$需要访问签名 Oracle 和随机 Oracle，$\mathcal{I}$需要自己回答这些 Oracle。为了简单起见，当$\mathcal{F}$询问一个消息的签名时，假定其已经对该消息进行了相应的 Hash 询问。如果没有，$\mathcal{I}$自己进行 Hash 询问。$\mathcal{I}$使用一个计数器 i，初始化为 0。

算法$\mathcal{I}$按照以下方式来回答伪造者$\mathcal{F}$对消息 m_i 的 Oracle 询问：

当$\mathcal{F}$对消息 m 进行 Hash Oracle 询问时，$\mathcal{I}$把计数器 i 增加 1，令 $m_i=m$，并随机选择 $r_i \in \mathbb{Z}_n^*$。$\mathcal{I}$依概率 p 返回 $h_i=r_i^e \bmod n$，依概率 $1-p$ 返回 $h_i=yr_i^e \bmod n$，这里 p 是一个固定的概率，其值将在后面确定。

当$\mathcal{F}$对消息 m 进行签名询问时，它已经对 m 进行了 Hash 询问，于是 m 等于某个 m_i。如果 $h_i=r_i^e \bmod n$，则$\mathcal{I}$返回 r_i 作为签名。由于 $h(m_i)=h_i=r_i^e \bmod n$，显然 r_i 就是 m_i 的有效签名。否则，算法中止，求逆算法失败。

最终，算法$\mathcal{F}$中止并输出一个伪造(m,σ)。假设$\mathcal{F}$之前已经对 m 进行了 Hash 询问。如果没有，$\mathcal{I}$自己进行 Hash 询问。无论何种情况，存在某个 i 使得 $m=m_i$。那么，如果 $h_i=yr_i^e \bmod n$，则有 $\sigma=h_i^d \bmod n=y^d r_i \bmod n$，于是$\mathcal{I}$输出

$$x = y^d \bmod n = \sigma / r_i \bmod n$$

作为 y 的逆。由于 $y \equiv x^e \bmod n$，x 即为所求。否则，算法中止，$\mathcal{I}$失败。

算法$\mathcal{I}$能够回答所有的签名询问的概率至少为 $p^{q_{sig}}$，然后$\mathcal{I}$输出 y 的逆的概率为 $1-p$。所以，$\mathcal{I}$至少以概率 $\alpha(p)=p^{q_{sig}}\cdot(1-p)$输出 y 的逆。容易看出，当 p 取$p_{max}=1-1/(q_{sig}+1)$时，$\alpha(p)$取最大值，并且

$$\alpha(p_{max}) = \frac{1}{q_{sig}}\left(1-\frac{1}{q_{sig}+1}\right)^{q_{sig}+1}$$

从而，有

$$\varepsilon(k) = \frac{1}{\left(1-\frac{1}{q_{sig}+1}\right)^{q_{sig}+1}} \cdot q_{sig} \cdot \varepsilon'(k)$$

而当 q_{sig}较大时，有 $\varepsilon \approx \exp(1)\cdot q_{sig}\cdot\varepsilon'$。

算法$\mathcal{I}$的运行时间等于算法$\mathcal{F}$的运行时间加上计算 h_i 所需要的时间，于是得到时间 t 的公式。

注意到 FDH 类型的签名方案是确定性的签名方案，每一个消息都有唯一的签名，所以在其安全性证明中，一个模拟器要么能够产生消息的签名，要么就不能产生其签名，这在一定程度上限制了安全性归约。这一发现推动了概率签名方法的设计思想，如 PSS 签名方法和 PFDH 签名方法，其中每一个消息都有多个签名。

3.3 标准模型下安全的数字签名和公钥加密方案

文献[2]、[3]是满足标准安全性证明的早期具有代表性的方案，实际上前面结果已经包含了许多这方面的内容，如一些基本概念等，这里简要介绍一些近年来的结果。

3.3.1 数字签名方案

文献[15]研究了数字签名方案中的 Hash 函数设计应用问题，降低了对 Hash 函数的要求；文献[16]提出了一类基于强 RSA 假设的数字签名方案。其共同之处是都不把 Hash 函数形式化为 RO。文献[16]的安全性证明实际上是用满足强计算假设的 Hash 函数取代了 RO。

1. 基于 padding 函数的 RSA 签名方案

克服 RSA 签名方案同态缺陷的一种通用解决方案就是先对消息应用 padding 函数作用，然后对结果作解密运算（签名）。文献[15]中的主要结果是：基于 padding 函数、对一组消息的 RSA 签名与对多个分组消息的 RSA 签名的安全性等价。而且这里并不要求 Hash 函数为 RO 或具有自由碰撞性质，只是假设存在某个安全的、用于签署固定长度消息的 padding 函数 μ；利用它就可以构造一个用于签署任意长度消息的安全 padding 方案。但该文在一般标准安全论断研究方面并未有多少进展。

2. 没有随机 Oracle 的安全签名方案

文献[16]基于强 RSA 假设（即对任意的 RSA 模 n，$s\in Z_n^*$，要在多项式时间内找到一个满足 $r^e\equiv s \bmod n$ 的二元组 $(e,r)(e>1)$ 是不可能的）提出了一种抵抗适应性选择消息攻击的签名方案，仍属 Hash-and-Sign 结构。密钥和参数说明类似于 RSA，注意公钥为 $n=pq$，$s\in Z_n^*$。签名算法本身很简单：$e=h(R,M)$，签名 σ 是 s 模 n 的 e 次根；验证算法略。

在安全性论断研究方面，文献[16]不再把 Hash 函数视为 RO，而是把 Hash 函数视为具有某些特定性质（如整除难处理性等）的随机函数 $h(R,M)$，这里 R 是随机因素。其特别之处在于：既充分利用 RO 安全论断的优点，又用一个假想的随机性 Oracle 取代 RO，即假设已知 h 的随机输入因素也对解决强 RSA 难题毫无帮助。文献[16]希望这种“相对模型方法论”能够替代 RO 方法论，但显然其假设过强（虽然文献[16]认为仍是现实的），归约也不“紧”，更重要的是目前看不到有推广应用的可能，文献[16]也承认这一点。

3.3.2 公钥加密方案

Cramer 和 Shoup[17] 于 1998 年提出了第一个比较实际的标准模型下可证明安全性的公钥加密方案，该方案的困难假设是判定性 Diffie-Hellman 问题。由于其安全性归约是在标准的 Hash 函数假设（抗碰撞）下得到的，并不依赖于随机预言模型，所以受到了很大的关注。

设 G 是有限域 Z_p^* 的阶为 q 的子群，p 和 q 为素数，且 $q\mid p-1$，g_1 和 g_2 是 G 中两个随

机的非单位元元素。设 $x=(x_1,x_2)$，$y=(y_1,y_2)$，$z=(z_1,z_2)$表示在 0 和 $q-1$ 之间的整数对；$g=(g_1,g_2)$，$u=(u_1,u_2)$表示 G 中的元素对；r 是 1 和 $q-1$ 之间的随机整数，记 $g^x=(g_1^{x_1},g_2^{x_2})$，$g^{rx}=(g_1^{rx_1},g_2^{rx_2})$。假设 H 是合适的抗碰撞 Hash 函数。

(1) 密钥生成算法：随机选取 $g_1,g_2\in G$，$x_1,x_2,y_1,y_2,z_1,z_2\in Z_q$，计算

$$c=g^x,\quad d=g^y,\quad h=g^z$$

用户 A 的私钥为$(x_1,x_2,y_1,y_2,z_1,z_2)$，公钥为$(g_1,g_2,c,d,h,H)$。

(2) 加密算法：为了发送消息 $m\in G$，选择一个随机数 r，令

$$u_1=g_1^r,\quad u_2=g_2^r,\quad e=h^r m,$$

然后计算

$$\alpha=H(u_1,u_2,e),\quad v=c^r d^{r\alpha}$$

密文就是四元组(u_1,u_2,e,v)。

(3) 解密算法：要解密(u_1,u_2,e,v)，用户 A 首先计算 $\alpha=H(u_1,u_2,e)$，然后利用他的私钥计算 $u^{x+\alpha y}$，并验证这个结果是否等于 v(因为 $u^{x+\alpha y}=g^{rx+r\alpha y}=c^r d^{r\alpha}$)。如果它不等于 v，拒绝该消息；如果通过这个检验，则继续进行解密：把 e 除以 u^z，因为 $u^z=g^{rz}=h^r$，而 $e=h^r m$，所以这就是明文 m。

对于 Cramer-Shoup 公钥加密方案，如果存在一个适应性选择密文攻击的敌手 $\mathcal{A}$ 能够破坏其语义安全性，那么就可以构造一个算法 $\mathcal{B}$ 来求解判定性 Diffie-Hellman 问题，即判断一个四元组(g_1,g_2,u_1,u_2)是否满足 Diffie-Hellman 性质 $\log_{g_1}u_1=\log_{g_2}u_2$。

定理 3.2 如果判定性 Diffie-Hellman 问题是难解的，那么 Cramer-Shoup 公钥加密方案在适应性选择密文攻击下是语义安全的。

证明 假设给定一个四元组(g_1,g_2,u_1,u_2)。首先算法 $\mathcal{B}$ 随机选取 x,y,z，令

$$c=g^x,\quad d=g^y,\quad h=g^z$$

并把(c,d,h)以及 $g=(g_1,g_2)$作为公钥发送给 $\mathcal{A}$。

$\mathcal{B}$ 需要回答 $\mathcal{A}$ 的解密询问，为了回答对(u_1',u_2',e',v')的询问，$\mathcal{B}$ 计算 $\alpha'=H(u_1',u_2',e')$，如果 $v'\neq(u')^{x+\alpha' y}$，则拒绝该消息，否则解密 $w'/(u')^z$ 返回给 $\mathcal{A}$。

当敌手 $\mathcal{A}$ 输出两个明文 m_0 和 m_1 用于区分测试时，$\mathcal{B}$ 随机选择 $b\in\{0,1\}$，设置密文为 $y^*=(u_1,u_2,e,v)$，其中

$$e=u^z m_b,\quad v=u^{x+\alpha y},\quad \text{而}\ \alpha=H(u_1,u_2,e)$$

敌手 $\mathcal{A}$ 需要判断 y^* 是 m_0 还是 m_0 的加密。

如果(g_1,g_2,u_1,u_2)满足 Diffie-Hellman 性质，那么 y^* 就是 m_b 的真实加密，其中 $r=\log_{g_1}u_1=\log_{g_2}u_2$ 就是加密所用的随机数，所以敌手 $\mathcal{A}$ 将以明显大于 1/2 的概率正确地猜测到 b 值。

如果(g_1,g_2,u_1,u_2)不满足 Diffie-Hellman 性质，那么 $\mathcal{A}$ 所看到的信息与 b 是无关的，所以他只能有 1/2 的机会猜测到 b 值。因此，通过使用不同的 x,y,z 值多次运行这个模拟攻击，如果 $\mathcal{A}$ 大多数情况下都能正确地猜测到 b 值，$\mathcal{B}$ 就确定(g_1,g_2,u_1,u_2)满足 Diffie-Hellman 性质。利用下面两个引理可完成归约证明。

引理 3.1 如果算法$\mathcal{B}$的输入是 Diffie-Hellman 四元组,那么敌手的视图(View,也称观察)与比特 b 的联合分布与实际攻击中的分布是不可区分的。

证明 设(g_1,g_2,u_1,u_2)是一个 Diffie-Hellman 四元组,考虑敌手的视图与比特 b 的联合分布。设 $u_1=g_1^r,u_2=g_2^r$。

在这种情况下,显然加密 Oracle 的输出分布是正确的。为了完成证明,需要论证解密 Oracle 的分布也是正确的。如果 $\log_{g_1}u_1'=\log_{g_2}u_2'$,则称$(u_1',u_2',e',v')$是一个有效的密文。

注意到,如果密文是有效的,而 $u_1'=g_1^{r'},u_2'=g_2^{r'}$,那么 $h_r^{r'}=(u_1')^{z_1}(u_2')^{z_2}$,因此,解密 Oracle 输出 $e'/h^{r'}$,这正是所期望的。根据下面的断言,即可证得引理 3.1。

断言:对密码体制的实际攻击和对模拟情况下的攻击,除了可忽略的概率之外,解密 Oracle 都拒绝所有无效的密文。

下面证明断言。根据敌手的视图,考虑四元组 $P=(x_1,x_2,y_1,y_2)\in\mathbb{Z}_q^4$的分布。设 $w=\log_{g_1}g_2$。

根据敌手的视图,P 是下面两个超平面相交的平面 P 上的一个随机点:

$$\log_{g_1}c=x_1+wx_2 \tag{3-1}$$

$$\log_{g_1}d=y_1+wy_2 \tag{3-2}$$

这两个方程容易由公钥推出。加密 Oracle 的输出并没有对 P 作进一步限制,因为

$$\log_{g_1}v=rx_1+wrx_2+\alpha ry_1+\alpha rwy_2 \tag{3-3}$$

假设敌手提交了一个无效的密文(u_1',u_2',e',v')给解密 Oracle,其中 $\log_{g_1}u_1'=r_1'$,$\log_{g_1}u_2'=wr_2'$并且 $r_1'\neq r_2'$。解密 Oracle 将会拒绝该密文,除非 P 恰好位于式(3-4)定义的超平面 H 上,即

$$\log_{g_1}v'=r_1'x_1+wr_2'x_2+\alpha'r_1'y_1+\alpha'r_2'wy_2 \tag{3-4}$$

其中 $\alpha'=H(u_1',u_2',w')$。注意到式(3-1)、式(3-2)和式(3-4)是线性无关的,H 与 P 相交于一条直线。

对于第一次提交的无效密文,解密 Oracle 将以 $1-1/q$ 的概率拒绝该密文,对于第 i 次提交的无效密文,解密 Oracle 将至少以 $1-1/(q-i+1)$的概率拒绝该密文。因此,除了可忽略的概率,解密 Oracle 将拒绝所有的无效密文。

引理 3.2 如果算法$\mathcal{B}$的输入是一个随机的四元组,那么比特 b 的分布与敌手的视图是无关的。

设 $u_1=g_1^{r_1},u_2=g_2^{wr_2}$,而且不妨假设 $r_1\neq r_2$,因为二者相等的概率是可忽略的。根据下面的两个断言,可证得引理 3.2。

断言 1:如果解密 Oracle 拒绝所有的无效密文,那么比特 b 的分布与敌手的视图是无关的。

为了证明这一断言,考虑点 $Q=(z_1,z_2)$。在攻击开始时,这是位于由公钥确定的直线

$$\log_{g_1}h=z_1+wz_2 \tag{3-5}$$

上的一个随机点。如果解密 Oracle 只对有效的密文(u_1',u_2',e',v')进行解密,则敌手得到的只是线性无关的关系 $r'\log_{g_1}h=r'z_1+r'wz_2$,不会泄露 Q 的信息。

下面分析加密 Oracle 输出的(u_1,u_2,e,v)。有 $e=\varepsilon m_b$，其中 $\varepsilon=u_1^{z_1}u_2^{z_2}$。考虑方程

$$\log_{g_1}\varepsilon = r_1 z_1 + w r_2 z_2 \tag{3-6}$$

显然，式(3-5)和式(3-6)是线性无关的，于是，相对于 b 以及除了 e 之外的敌手的视图，ε 是均匀分布的。换言之，ε 是一个完善的一次一密。因此 b 与敌手的视图是无关的。

断言 2：除了可忽略的概率，解密 Oracle 拒绝所有的无效密文。

为了证明断言 2，与引理 3.1 一样，在敌手视图的条件下考虑 $P=(x_1,x_2,y_1,y_2)$的分布。根据敌手的视图，这是位于超平面方程(3-1)、(3-2)和下面的超平面交叉的直线上的一个随机点：

$$\log_{g_1}v = r_1 x_1 + w r_2 x_2 + \alpha r_1 y_1 + \alpha r_2 w y_2 \tag{3-7}$$

方程(3-7)来自于解密 Oracle 的输出。

假设敌手提交了一个无效密文$(u_1',u_2',e',v')\neq(u_1,u_2,e,v)$，其中

$$\log_{g_1}u'_1 = r'_1,\quad \log_{g_1}u'_2 = r'_2,\quad r'_1 \neq r'_2$$

设 $\alpha'=H(u_1',u_2',e')$。

考虑下面 3 种情况。

情况 1：$(u_1',u_2',e')=(u_1,u_2,e)$。

在这种情况下，Hash 值是相同的，但是 $v'\neq v$，所以解密 Oracle 当然会拒绝它。

情况 2：$(u_1',u_2',e')\neq(u_1,u_2,e)$，$\alpha'\neq\alpha$。

除非点 P 位于方程(3-4)定义的超平面 H，解密 Oracle 将拒绝。但是，方程(3-1)、(3-2)、(3-4)和(3-7)是线性无关的，这可以通过下面式子进行验证

$$\det\begin{bmatrix} 1 & w & 0 & 0 \\ 0 & 0 & 1 & w \\ r_1 & wr_2 & \alpha r_1 & \alpha w r_2 \\ r'_1 & wr'_2 & \alpha' r'_1 & \alpha' w r'_2 \end{bmatrix} = w^2(r_2-r_1)(r_2'-r_1')(\alpha'-\alpha)\neq 0$$

因此，H 与直线 L 相交于一点，由此可知解密 Oracle 拒绝，除了可忽略的概率之外。

情况 3：$(u_1',u_2',e')\neq(u_1,u_2,e)$，$\alpha'=\alpha$。

这一情况发生的概率是可忽略的，否则与 Hash 函数的抗碰撞性质矛盾。

3.4 面向会话密钥分配协议的安全模型及其应用

通信双方在充满敌意的环境中传送数据，一般需要确保数据的机密性和可认证性。要达到此目的，必须加密和认证被传送的数据，这就需要密钥，而密钥通常需要通过会话密钥分配(SKD)协议来实现。当前最常见的是 3 方 SKD 协议(可信方参与)，因此下文的论述以此为重点。

最早、最流行的 3 方密钥分配系统是 1978 年提出的 NS 系统[18]，并且有许多具体候选方案。之后的数年，又有 10 多个 SKD 协议出现。但不幸的是，似乎所有这些工作都存在这样的“怪圈”：提出一个协议；然后是不断地试图破译；不断地修补。实际上 Needham 和 Schroeder 在一开始就提出了警告：这样开发的协议容易有微妙的弱点，且不易在正常操作

中检测到，很有必要研究验证协议正确性的技术。作为对这种警告的证实，文献[19]指出了一种 NS 协议的 bug，许多相关协议都有类似的缺陷。如此漫长的攻击历史使得人们终于达成这样的共识：要解决会话密钥分配问题，仅仅由作者给出一个协议、并且作者本人找不到可行的攻击手段是远远不够的。

Burrows、Abadi 和 Needham[20]试图通过使用特定目的的逻辑来解决这个问题，即著名的 BAN 逻辑，相关研究目前仍在继续，相关理论与方法将在下一章介绍。形式化逻辑方法在寻找 bug 方面很有成效，且有些方法可以自动化；但问题在于，一旦抽象的密码运算实例化，“逻辑正确”的证明并不意味着协议本身必然是正确的，也就是说，缺乏严格的安全性证明。

使用可证明安全性理论来研究 SKD 协议最初是由 Bellare 和 Rogaway[4]发起的。他们不但定义了本原的安全性，还定义了目标安全性，也有助于找到 bug。更重要的是，SKD 协议的安全性可以得到证明。这项研究在实现基本安全目标方面已经取得了巨大成功，存在的问题是，随着附加越来越多的目标，定义和证明的复杂性大大增加了。一个发展方向是把可证明安全性和形式化分析方法结合起来，这就是通常所说的混合方法。

SKD 的可证明安全性研究主要包括以下 3 个方面的内容。

(1) 定义。直接给出安全性定义，再证明符合定义。

(2) 可信模型。如 2 方基于对称密码的模型，3 方模型等。

(3) 安全目标。主要有认证、新鲜性、机密性、已知密钥攻击、前向机密性（Forward Secrecy）及字典攻击等。

需要说明的是，作为一种较新的安全目标，前向机密性是指，在 SKD 协议结束、当前会话密钥产生后，即使这时敌手得到了任何一方的主密钥，也不能得到以前会话密钥的任何特定信息。其现实背景是：在实际应用中，计算机系统比协议的安全性差一些，前向机密性缓减了由于系统遭受入侵带来的损失。

3.4.1 BR 安全模型及其应用

1. BR 安全模型

Bellare 和 Rogaway 在 1993 年和 1995 年分别给出两个可证明安全性的 2 方和 3 方 SKD 协议[21,22]。协议本身很简单，其重要意义在于首次建立了 SKD 协议的形式安全模型，称为 BR 安全模型。这是一种现实模型，亦即协议只定义在现实世界中，而安全性是通过会话密钥与某随机数的计算不可区分性来定义的（基本方法是使会话密钥和一个敌手易于得到的某伪随机数紧密联系）。因为原理类似，下面以 3 方模型为例作一简要介绍。

(1) 协议参与方。$I=\{0,1,2,\cdots,N\}$（0 代表可信中心 S），敌手 E。

(2) 协议定义。$P=(\Pi,\psi,LL)$，这是一个多项式时间可计算的三元组函数，Π 描述诚实方的行为，ψ 描述 S 的行为，LL 描述用户主密钥的初始分布，具体说明可参见参考文献[22]。

(3) 敌手模型。E 控制所有合法方之间的通信（如可以控制协议启动时间、篡改、替换或删除数据等），形式化为一个概率图灵机，具有 Oracle $\Pi_{i,j}^{s}$ 和 $\psi_{i,j}^{s}$，$\Pi_{i,j}^{s}$形式化了成员 i 试图

和成员 j 协商一个会话密钥的通信实例 s，$\psi_{i,j}^{s}$ 形式化了 S 试图给 i，j 分配会话密钥的通信实例。E 所能发起的攻击形式化为 5 种 Oracle 询问：(SendPlayer，i，j，s，x)；(SendS，i，j，s，x)；(Reveal，i，j，s)；(Corrupt，i，K)；(Test，i，j，s)。前 4 种 Oracle 询问可以是多项式次，形式化了 E 所能发动的各种攻击，如窃听、已知会话密钥、重放、收买某合法方等攻击，(Test，i，j，s)则只能询问一次，是为了定义安全性而提供的"测试"Oracle。

(4) 安全性定义。除合理性(Validity)之外，如果敌手得到 Oracle 询问(Test，i，j，s)的回答后，对挑战(Challenge)"猜测"正确的优势函数 $\mathrm{Adv}_{P,f,S_n}^{E}(k)(=2\Pr[\mathrm{Good}-\mathrm{Guess}]-1)$是可忽略的，就称协议是安全的。这里当敌手发起 Test 询问时，Oracle 回答如下：$b\in_R\{0,1\}$，如果 $b=0$，返回一个随机数，否则返回敌手要得到的会话密钥；如果敌手对 b 的取值猜测正确，就称事件"Good-Guess"发生，敌手成功。上述定义是说，如果敌手的成功概率只以可忽略优势偏离 1/2，敌手失败，即协议是安全的。

2. BR 安全模型的应用

人们基于 BR 安全模型设计了大量的 SKD 协议，有代表性的 SKD 协议可参见文献[21]～[24]。文献[21]提出了一种 2 方 SKD 协议，该协议是一种基于非对称密码方案(公钥加密、签名)的 SKD 协议。该协议具有以下性质：如果公钥加密和签名方案满足正规的、易理解的安全目标，则协议抗已知会话密钥攻击安全，具有语意安全性(Semantic Security，即敌手得不到会话密钥 K 的任何信息)。

文献[22]提出了一种 3 方 SKD 协议，概述如下。

(1) $A\rightarrow B$：R_A。

(2) $B\rightarrow S$：R_A，R_B。

(3) $S\rightarrow A$：$C_A=\mathrm{Enc}_{a[1]}(K)$，$\mathrm{Mac}_{a[2]}(A,B,R_A,C_A)$。

(4) $S\rightarrow B$：$C_B=\mathrm{Enc}_{b[1]}(K)$，$\mathrm{Mac}_{b[2]}(A,B,R_B,C_B)$。

这里 $a[]$、$b[]$表示主密钥，这是基于对称密码方案(对称加密，认证码)的 SKD，可证明安全性质同上。

以上协议的缺点是不具有"前向机密性"，Bellare 等又提出了一个具有"前向机密性"的 SKD 协议[23]，该协议和具有认证功能的 Diffie-Helllman 协议基本相同，区别只在于对会话密钥用 Hash 函数做了随机化处理：$K=H(g^{xy}\bmod p)$，可证明安全性是基于 RO 模型的。

此外，Bellare 和 Pointcheval 等在基于口令的 SKD 协议研究方面也取得了很好的结果，在 2000 年提出的一种兼具"前向机密性"和抗字典攻击性质的 SKD 协议[24]，安全性证明是基于 RO 模型中的 Diffie-Hellman 问题。基于口令的 SKD 协议是一种很有应用前景的协议，后面将详细论述。

对 SKD 协议主要有两类攻击：其一是利用基础本原的弱点发动攻击，如破译分组密码或伪造 MAC；其二是利用协议设计的缺陷发动攻击，比如对 NS 协议的已知密钥攻击。而可证明安全性的目标是证明不能存在源于协议缺陷的攻击，暗含假设是，不存在针对基础本原的攻击，亦即基础本原具有清晰的、标准的、易于理解的密码性质。

BR 安全模型作为第一个有关 SKD 协议的形式化安全模型得到了广泛的重视，在实践

中不断得到改进。然而它只定义在现实世界中，因此归约过程较为烦琐，可操作性稍差。研究定义在两个世界(理想和现实世界)中的协议模型就成为一种很好的选择(因为理想模型中的协议和现实中的协议之间的转换更易于操作和理解)，最具有代表性的结果就是著名的BCK 安全模型[25]。

3.4.2　BCK 安全模型及其应用

文献[25]中给出了构造和分析 SKD 协议的一般框架，给出了正确的形式化方法，并建议在设计复杂协议时采用简单、富有吸引力的模块化设计原则，称为 BCK 安全模型。BCK 安全模型的基本思想是：基于模块化观点，首先把 SKD 协议定义在理想模型中，然后像 BR 安全模型那样利用计算不可区分性来定义理想模型中的安全性，最后协议被“编辑”成现实模型中的协议；归约证明时要求现实敌手必须能“模仿”理想敌手，所谓“模仿”概念也就是不可区分性。

1. BCK 安全模型简介

该模型适用于研究认证通信问题，并特别强调相关的密钥交换问题。

(1) 敌手模型。敌手控制合法用户的通信信道，可以修改或删除传送消息，甚至可以插入假消息；还控制了消息发送的延迟；可能还具有额外的能力，如收买用户(很多时候，这模拟了通过入侵用户计算机系统获得用户秘密参数)。认证链模型中的敌手称为 **AM 敌手**，概括说来除了收买某方的情形，AM 敌手不能伪造消息，只是忠实地传递消息(虽然可以改变传递顺序、延迟等)；非认证链模型中的敌手称为 **UM 敌手**，能力要强得多。

(2) 安全目标。确保传送数据的可认证性。简单采用签名或 MAC 可能是不够的(当然它们通常是设计解决完整性办法的基础)，因为网络本质上是异步的，协议经常是“消息驱动”的。

(3)“通用编译器”C。这是模块化研究的核心组件，作用是把理想认证模型中的任何协议 π 转换成现实协议 $\pi'=C(\pi)$，后者完成和前者一样的任务，但能够抵抗强得多的现实敌手。认证器(Authenticator)是一个特殊的编译器 C：对任何协议 π，$C(\pi)$在非认证网络中模仿 π。设计认证器通常可以归约为更简单的协议，如 MT-authenticator(消息传播认证器)，其目标只是认证用户间简单的消息交换，可以基于简单的密码函数(如 MAC、数字签名、公钥加密)来构造它。认证器的定义涉及认证和非认证模型的形式化定义以及运行在两个模型中的协议的等价概念，后者的基本要素是一个协议被另一个“模仿”的概念，它源于安全多方计算协议的一般定义[26]。

(4) 协议定义。设计与分析在非认证网络中的协议可划分为两个独立的阶段：首先在认证模型中设计并证明协议安全性；然后应用特定的认证器确保被“编译”后的协议在非认证环境中，维持和理想认证模型中相同的行为。这样大大简化了设计与分析工作，还为设计者提供了一种“debug 工具”，帮助消除不必要的协议元素。具体而言，SKD 协议定义为一种**“消息驱动协议”**：它是这样的一个迭代进程，具有某初始状态(协议的输入、随机输入，身份)的某方调用协议后，协议等待激活(Activation)：可由两类事件引起，来自网络的消息到

达或一个外部请求(这形式化了来自该方运行的其他进程的信息);激活后,协议根据输入数据、当前内部状态,产生一个新的内部状态、一个向网络发出的消息、一个给该方运行的其他协议(或进程)的外部请求,此外还产生一个输出(它是累加的,即开始为空,每次激活后的输出被附加上去);激活结束后,协议等待下一次激活。协议可以形式化表示为一(概率)函数:

π(当前状态,接收消息,外部请求)=(新状态,发出消息,发出请求,输出)在认证链模型 AM(Authenticated-Link Model)或非认证链模型中激活都由对应敌手 A 控制和编排。协议的整体输出即各方包括敌手的累加输出的连接,这里敌手的输出是敌手观察(Adversary View)的函数。敌手观察是指:敌手在整个计算期间,利用自己的随机输入,看到或推导出的信息。

(5) 协议的模仿。设 π、π'是 n 方消息驱动协议,称 π'在非认证网络中模仿 π,如果对任何 UM 敌手 U,存在 AM 敌手 A,使得对任何输入 x,满足

$$\mathrm{Auth}_{\pi,A}(x) \overset{c}{\approx} \mathrm{Unauth}_{\pi',A}(x)$$

上式表示计算不可区分。

这里 $\mathrm{Auth}_{\pi,A}(x,r)=\mathrm{ADV}_{\pi,A}(x,r),\mathrm{Auth}_{\pi,A}(x,r)_1,\cdots,\mathrm{Auth}_{\pi,A}(x,r)_n$,是指在认证链模型中,根据输入 x 及随机输入 r 与敌手交互运行协议 π 之后,敌手以及全部 P_i 的累加输出,$\mathrm{Unauth}_{\pi,A}(x)$的说明类似,但限于非认证链模型。

综上可见,认证器把(某种良好定义意义上)在认证链模型中安全的协议转化成非认证链模型中的安全协议,无疑构造认证器至关重要。

MT 认证器主要用于设计认证器(Authenticator):首先设计一个"低层"协议 λ,接收外部要求发送消息的请求,然后以认证的方式发送这些消息;其次,已知某协议 π(在认证链模型中工作),认证器输出一个协议 π',与 π 只有一个区别——消息要经过 λ 传递,也就是说,发出的消息不再直接传给网络,而是激活 λ 去传递消息;不再直接由网络接收消息,而是取自 λ 的输出。显然 MT 认证器 λ 在非认证网络中模仿认证网络中的消息传送协议(MT)。因此可以定义以下编译器。

编译器 C_λ:给定协议 π,生成 $\pi'=C_\lambda(\pi)$在 P_i 范围内运行,首先调用 λ;对 π 发送的每一消息,π'用发送该消息给预定收方的外部要求激活 λ;每当 π'被某接收消息激活,都用它激活 λ,当 λ 输出"P_i 从 P_j 处收到 m",π 就被来自 P_j 的 m 激活。

定理 3.3 设 λ 是 MT 认证器,C_λ 是基于 λ 的编译器,则 C_λ 是一个认证器。

文献[25]中还指出,MT 认证器作为最基础模块,可以根据基本的密码学工具构造,如可以根据公钥签名构造如图 3.1 所示的 MT 认证器 λ_{sig}。

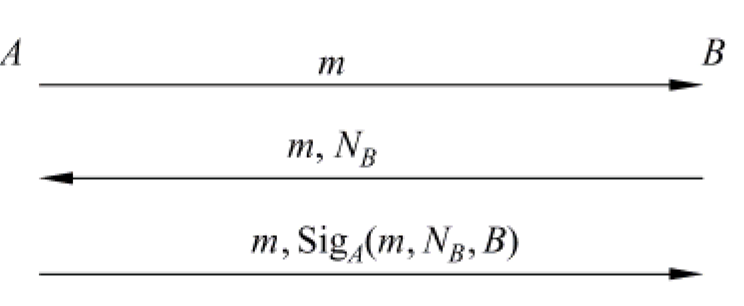


图 3.1 MT 认证器

定理 3.4 假设签名方案是抗选择消息攻击安全的,则 λ_{sig}在非认证网络模仿协议 MT。

2. BCK 安全模型的应用

因有以下结论,无妨重点研究认证链模型中的 SKD 协议(这时敌手实际是被动的)。

定理 3.5 如果 π 是认证链模型中的安全 SKD 协议,C 是认证器,则 $C(\pi)$是非认证链

模型中安全的 SKD 协议。

定义安全 SKD 协议：首先规定理想进程——表示了凭直觉对 SKD 协议所能期望的最好特征；称一个 SKD 协议（在认证或非认证链模型中）是安全的，如果它能模仿对应理想进程。

(1) 理想 SKD 进程的形式化。存在 n 方 $P_1, \cdots, P_n$，以及一个理想敌手 S，假设还有一个可信方 T。计算过程包括一系列由 S 导致的激活，共有以下 4 类激活。

① 调用 P_i，以便和 P_j 建立一个新密钥。最终效果是数值“P_i 和 P_j 建立了一个密钥 (K,s)”被加到 P_i 的输出，这里 K 是按预定分布选择的密钥，s 包括身份、序号等信息。S 仅能得到 s，不知道 K（实际上可想象由 T 把 K 传给 P_i。如果建立密钥的双方有一方被收买，规定由敌手选择密钥）。

② 调用 P_j，以便和 P_i 建立会话 s 的密钥。和以上说明类似，但 R 规定 P_j 是应答方（可想象由 T 把 K 传给 P_j）。

③ 收买会话 s。这种激活仅当 s 当前是合法的；效果是敌手得到了对应会话 s 的密钥 K，此外“会话 s 被收买”被加到发起方的输出上。

④ 收买 P_i。效果是 P_i 知道的所有密钥都为敌手所知，此外“P_i 被收买”被附于 P_i 的输出。

(2) 理想进程的整体输出。各方包括敌手的累加输出的连接。

$\mathrm{Ideal}_S(r_S, r_T) = \mathrm{ADV}_S(r_S, r_T), (\mathrm{Ideal}_S(r_S, r_T))_1, \cdots, (\mathrm{Ideal}_S(r_S, r_T))_n$，具体说明与前面类似。

定义 3.7（两种模型中安全的 SKD 协议） 设 π 是 n 方消息驱动协议，称 π 在非认证网络中是安全的 SKD 协议，如果对任意 UM 敌手 U，存在一个理想敌手 S，使得

$$\mathrm{Unauth}_{\pi,U} \overset{c}{\approx} \mathrm{Ideal}_S()$$

称 π 在认证网络中是安全的，如果对任意 AM 敌手 A，存在一个理想敌手 S，使得

$$\mathrm{Auth}_{\pi,A} \overset{c}{\approx} \mathrm{Ideal}_S()$$

就具体应用而言，基于以上理论和必要的安全假设（DLP）不难证明，著名的原始 Diffie-Hellman（DH）协议（图 3.2）就是认证链模型中安全的密钥交换协议，然后基于前面介绍的认证器 λ_{sig} 不难转化成现实模型中的安全 SKD 协议。不难看出，这也正是人们出于对中间人侵攻击的考虑对 DH 协议的改进，即认证 Diffie-Hellman 密钥协议，而 BCK 模型理论恰好证明了这种改进的安全性。

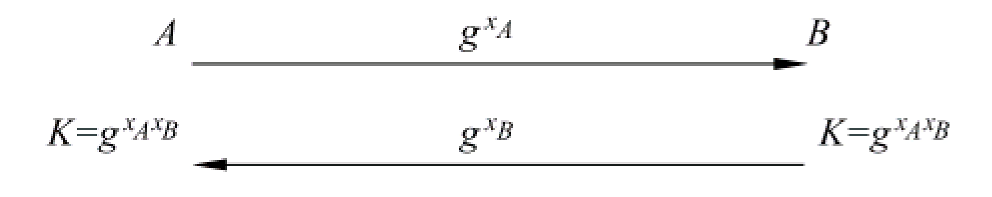


图 3.2　原始 Diffie-Hellman 协议

3.4.3　PSKD 安全模型

前两节介绍的 SKD 协议通常依赖于 PKI 技术或通信双方共享高品质秘密密钥（主密

钥)。问题是完全应用公钥技术成本较高,而对称密码算法的主密钥长度一般较长,显然不易记忆。因此一个自然的思路就是各方共享易于人类记忆的口令,由它构造出较高品质的会话密钥,当然因上述口令可能取自一个较小的字典(多项式级),易受离线字典攻击,必须慎重评估其安全性。

基于口令的 SKD 协议(简记为 PSKD)的安全目标是:任何 PPT 敌手的成功概率上界为 $O(t/|D|)+\mu(n)$,这里 t 是敌手发动的会话数目,$\mu()$是安全参数的可忽略函数,$D\subseteq\{0,1\}^n$ 是字典。

第一个真正意义上的 PSKD 协议是由 Bellovin 和 Merritt 提出的[27],这一工作影响深远,已成为该领域的研究基础,但该协议的安全性并未得到严格的论证,仅仅具有基于假想论断的推测安全性。第一个对此进行严格研究的是 Halevi 和H. Krawczyk[28],他们实际上是考虑了一种非对称混合(Hybrid)模型:一方(Server)可持有一个高质量密钥,其他方仅持有口令;任何人都可以安全地得到服务器的公钥。Halevi-Krawczyk 模型适用于非对称访问控制环境,但对于两个人之间建立通信则完全不适用,而且要求人在访问控制环境中持有不易记忆的公开口令也不现实。那么是否可能实现一个仅基于易记口令的安全访问控制机制和认证密钥交换协议呢?

首先对上述问题作出肯定回答的是文献[29]和[30]。文献[29]和[30]的主要贡献是具体给出了两个在 RO 模型中可证明安全的基于口令的会话密钥分配协议。在标准模型中可证明安全的方案参见文献[31]和[32],文献[31]基于一次签名技术提出的 PSKD 协议称为 KOY 协议,安全性基于 DDH(确定性 Diffie-Hellman)假设。文献[33]则把 KOY 协议进一步推广到门限认证机制。Goldreich 则提出了第一个(也是至目前为止唯一一个)仅基于标准密码假设——“陷门置换存在”的可证明安全 PSKD 协议[32]。其核心思想是:一方首先选择一个在较大的域上的线性多项式 Q,与另一方执行一个安全的多项式赋值算法(安全多方计算),$Q(w)$即会话密钥,这里 w 是口令。

文献[32]也采用了“协议模仿”观点进行安全性证明。通俗地讲,敌手对以上协议的攻击效果可以类比为这样的攻击:敌手只被允许询问常数次“w 是 A 的口令吗”的问题。该协议的缺陷是实现效率仍然不理想,而且不支持一对用户的并行操作。但有着较深远的理论意义。

下面主要来介绍 PSKD 安全模型。

基本约定:通信双方 A、B;信道 C 即 PPT 敌手,具有 Oracle$A$,B,$C^{A(x),B(y)}(\sigma)$是 C 以 σ 作为辅助输入,与分别以 x、y 为输入的 A、B 通信时的输出;口令字典 $D\subseteq\{0,1\}^n$,可以 PPT 取样,$\varepsilon=1/|D|$,$|D|$可以表示成 n 的函数 $m(n)$。

定义 3.8((1-ε)不可区分) 设$\{X_n\}$和$\{Y_n\}$是两个概率空间,称它们是$(1-\varepsilon)$不可区分的,表示为$\{X_n\}\overset{\varepsilon}{\equiv}\{Y_n\}$,如果对任意 PPT 算法 A 及所有辅助输入 $z\in\{0,1\}^{\mathrm{poly}(n)}$,$|\Pr[A(X_n,1^n,z)=1]-\Pr[A(Y_n,1^n,z)=1]|<\varepsilon+\mu(n)$。其中 $\mu(n)$是一个可忽略函数。

注意:如果取 ε 为指数函数,就得到通常的计算不可区分概念。因此,定义 3.8 是计算不可区分概念的推广。也称定义 3.8 为弱计算不可区分。类比伪随机性理论[1],给出以下

的推广定义。

定义 3.9((1−ε)伪随机性)　称$\{X_n\}$是(1−ε)伪随机的,如果它与均匀空间$\{U_n\}$是(1−ε)不可区分的。

也称定义 3.9 为弱伪随机性。

(1) 基本思想。采用“模仿”论断,即使得运行于现实模型的协议模仿理想模型的功能,两种情况下的输出不可区分。

(2) 理想模型。A'、B'是共享口令$w\in_R D$的诚实方,C'是任意 PPT 理想模型敌手,有辅助输入σ;A'、B'把口令交给可信第三方,C'发送 0 或 1(接受/拒绝),若C'发送 1,$k\in_R\{0,1\}^n$,发给A'、B';若C'发送 0,$k\in_R\{0,1\}^n$,k被发给A'、B'则收到符号⊥(失败符号),两种情况下C'未收到输出。

(3) 理想分布。$\text{Ideal}_{C'}(D,\sigma)=(w,\text{output}(A'),\text{output}(B'),\text{output}(C'(\sigma)))$,即如果$C'$发出 1,则$\text{Ideal}_{C'}(D,\sigma)=(w,U_n,U_n,\text{output}(C'(\sigma)))$,否则,$\text{Ideal}_{C'}(D,\sigma)=(w,U_n,\bot,\text{output}(C'(\sigma)))$。

(4) 现实模型。A,B说明同前,C是任意 PPT 现实模型敌手亦即信道,有辅助输入σ;初始化同前,协议执行表示为$C^{A(x),B(y)}(\sigma)$,C有一个决定比特,决定某一方的私有输出是密钥或⊥(失败符号),无妨设是B;现实分布为

$$\text{Real}_C(D,\sigma)=(w,\text{output}(A),\text{output}(B),\text{output}(C^{A(w),B(w)}(\sigma)))$$

(5) 协议安全性定义。称一个 PSKD 协议是安全的,如果满足以下两个条件。

① 被动敌手:任给 PPT 现实敌手C,存在 PPT 理想敌手C'(总发送 1),满足

$$\{\text{Ideal}_{C'}(D,\sigma)\}_{n,D,\sigma}\overset{c}{\equiv}\{\text{Real}_C(D,\sigma)\}_{n,D,\sigma}$$

② 主动敌手:任给 PPT 现实敌手C,存在 PPT 理想敌手C',满足

$$\{\text{Ideal}_{C'}(D,\sigma)\}_{n,D,\sigma}\overset{O(\varepsilon)}{\equiv}\{\text{Real}_C(D,\sigma)\}_{n,D,\sigma}$$

这里$\varepsilon=1/|D|$。

上述安全性定义有以下特点:输出会话密钥是$(1-O(\varepsilon))$伪随机的;具有前向机密性,即口令泄露不影响当前会话密钥的安全,抗已知密钥攻击;具有入侵检测特性,即敌手对任何会话消息的修改,至少以概率$(1-O(\varepsilon))$被检测出。

定理 3.6　假设存在单向陷门置换,则存在安全的(强化)PSKD 协议。

这是文献[32]的主要结果,证明完全是构造性的。

3.5　基于口令的安全协议的模块化设计与分析

基于口令的安全协议的突出优点是:口令短,易于人类记忆和选取,省去了很多额外装置(如 IC 卡)的开销。但从安全性角度看,与一般的基于高品质主密钥或公钥的安全协议相比,基于口令的安全协议也面临着严峻的挑战,最突出的问题就是抗字典攻击问题。

除了 3.4.3 节提到的关于 PSKD 方面的研究成果外,文献[34]中提出了一个新的基于口令的安全协议,称之为口令更换(Protected Password Change,PPC)协议。事实上,文献

[34]中的 PPC 协议可以追溯到文献[35]中的研究结果。这类协议的提出背景是,鉴于口令字通常长度较短,因此基于口令的安全机制存在许多需要遵循的特殊准则,其中之一就是要定期更换口令以确保口令的安全性。然而文献[36]中的研究结果表明,该 PPC 协议[34]易于遭受字典攻击和拒绝服务攻击,因此是不安全的。文献[36]中主要利用了这样一个事实:与普通安全协议相比,PPC 协议中通信双方共享的旧口令 pw 和准备更换的新口令 pw' 都是可以穷举的(均取自一个多项式数量级字典),通过截取并分析通信双方共享信息,可以推导出 pw 和 pw' 的关联等式,利用离线字典攻击即可破解新、旧口令;此外,该 PPC 协议[34]的认证措施也存在缺陷。

由此可知,基于口令的安全协议不可能期望达到和基于高品质主密钥或公钥的安全协议同等的安全性,对基于口令的安全协议,敌手总可以实施在线字典攻击(口令猜测),即敌手仅仅从口令字典 D 中随机选取一个猜测值去假冒合法用户执行协议,成功概率至少是 $1/|D|$;经过 t 次假冒攻击,成功概率至少是 $t/|D|$。因此,一个安全的基于口令的安全协议应该能够限制敌手只能发动以上的在线猜测口令攻击,而成功概率不会以不可忽略优势偏移以上成功概率。具体来说,基于口令的安全协议的基本安全目标如下。

(1) 敌手(Adversary)发动离线字典攻击的成功概率是可忽略的。

(2) (主动)敌手发动在线字典攻击(如假冒攻击)的成功概率为 $O(1/|D|)+\mu(n)$,这里 $\mu(n)$ 是一个可忽略函数。

权衡基于口令的安全协议的众多优点,上述安全性定义在具体应用中是允许的:通常在线通信次数必然是多项式时间的,而且会远远小于口令字典的基数。就实际应用而言,敌手的全部可能在线猜测只能排除很少的候选口令。如果再辅以其他措施,如定期更换口令,则基于口令的安全协议在应用安全性方面不会有什么问题。

但从计算复杂性理论角度看,对于基于口令的安全协议来说,敌手的成功概率毕竟是多项式而非指数函数的倒数,并非可忽略函数,这与一般基于高品质主密钥或公钥的安全协议存在很大区别。从理论上深入研究这一点,对安全协议的安全性的具体影响是非常有意义的,但目前尚无文献深入研究基于口令的安全协议的理论基础。

文献[37]在系统分析相关研究结果的基础上,借鉴文献[32]中的思想,以“弱计算不可区分”概念为基础重点研究了基于口令的安全协议的理论基础——“弱伪随机性”理论。然后将文献[38]中提出的安全协议模块化设计与分析的观点推广到基于口令的安全协议的设计与分析,即系统给出了基于口令的安全协议的模块化设计与分析理论。最后,利用上述理论工具,具体给出了两类实现简单快捷、具有可证明安全性的安全协议,即 PSKD(基于口令的会话密钥分配)协议和 PPC(口令更换)协议。本节主要介绍文献[37]中的相关研究成果。

3.5.1 基于口令的安全协议的理论基础——弱伪随机性理论

要建立基于口令的安全协议的模块化设计与分析理论,首先必须明确基于口令的安全协议的理论基础。下面以弱伪随机性概念为中心讨论这个问题。

1. 基本概念

可证明安全性理论的一般基础假设:单向函数或单向置换是存在的。这是任何安全协

议的安全性基础。

定义 3.10(弱伪随机生成器) $(1-1/m(n))$伪随机生成器是一个满足以下条件的确定性多项式时间算法 G。

(1) 扩展性:存在扩展因子 $l(n)>n$,$|G(s)|=l(|s|)(s\in\{0,1\}^*)$。

(2) 弱伪随机性:概率空间$\{G(U_n)\}_{n\in N}$是$(1-1/m(n))$伪随机的。

定义 3.11(弱单向函数) 称函数 $f:\{0,1\}^*\rightarrow\{0,1\}^*$ 是弱单向的,如果满足下面的条件:

(1) f 是多项式时间可计算的。

(2) 存在多项式 $p(\cdot)$,使得对任意 PPT 算法 A,对所有足够大的 n,有

$$\Pr[A(f(U_n))\notin f^{-1}(f(U_n))]>1/p(n)$$

通俗地讲,弱单向函数是指求逆失败概率"不太小"(与单向函数相比)的函数。显然,单向函数必然是弱单向函数,但反之不然,实际上存在大量不是单向函数的弱单向函数[1]。

定义 3.12(ε 单向函数) 称函数 $f:\{0,1\}^*\rightarrow\{0,1\}^*$ 是 ε 单向的,如果满足下面条件:

(1) f 是多项式时间可计算的。

(2) 对任意 PPT 算法 A,对所有足够大的 n,有 $\Pr[A(f(U_n))\notin f^{-1}(f(U_n))]>\varepsilon$。

显然弱单向函数就是 $1/p(n)$单向函数。另外根据文献[1],存在由弱单向函数构造单向函数的一般 PPT 算法。

定义 3.13(统计差异) 两个概率空间$\{X_n\}$和$\{Y_n\}$的统计差异(Statistical Difference)定义为

$$\Delta(n)=1/2\sum_{\alpha}|\Pr[X_n=\alpha]-\Pr[Y_n=\alpha]|$$

统计差异衡量了两个统计分布曲线的接近程度。

2. 基本性质

定理 3.7 如果对于任意的 $p(n)$,$\Delta(n)<1/m(n)+1/p(n)$,则 X、Y 是弱计算不可区分的。

证明 设 f 是任意布尔函数(PPT 算法),令

$$S_f=\{x:f(x)=1\}$$

则有

$$1/m(n)+1/p(n)>\Delta(n)=\max_S|\Pr[X_n\in S]-\Pr[Y_n\in S]|$$
$$\geqslant|\Pr[X_n\in S_f]-\Pr[Y_n\in S_f]|=|\Pr[f(X_n)=1]-\Pr[f(Y_n)=1]|$$

由 f 的任意性以及统计差异的定义即可得证。

注意定理 3.7 的逆命题一般不成立。

以上的弱计算不可区分定义是针对单个统计样本而言,但在具体应用中经常涉及对多重样本的统计分析,如对序列密码的多个截断序列或分组密码的多个明/密文对取样分析,特别是任意 PPT 敌手对基于口令的安全协议做多次在线猜测攻击得到的"观察(View)"样本。因此,分析重复取样条件下的弱伪随机性质保持程度是很有意义的。

以下讨论表明,与伪随机序列相比,弱伪随机序列的多项式取样不能在完全意义下保持

弱不可区分性质，这一点和伪随机序列的多项式取样完全不同。

定义 3.14(重复取样的弱计算不可区分性) 称任意两个概率空间 $X=\{X_n\}_{n\in N}$，$Y=\{Y_n\}_{n\in N}$是多项式时间取样$(1-1/m(n))$计算不可区分的，如果对任意 PPT 算法 A，任意多项式 $q(\cdot)$，$p(\cdot)$，对足够大的 n，有

$$|\Pr[A(X_n^{(1)},\cdots,X_n^{q(n)})=1]-\Pr[A(Y_n^{(1)},\cdots,Y_n^{q(n)})=1]|<1/m(n)+1/p(n)$$

这里$\{X_n^{(i)}\}$、$\{Y_n^{(i)}\}$分别是独立同分布的随机变量序列。

定理 3.8 设任意两个概率空间 $X=\{X_n\}_{n\in N}$，$Y=\{Y_n\}_{n\in N}$都是多项式时间可构造的，如果 X、Y 是$(1-1/m(n))$计算不可区分的，则其多项式$(q(n))$取样序列是$(1-q(n)/m(n))$计算不可区分的(注意根据具体应用背景，$q(n)<m(n)$属于自然限制)。

证明 采用著名的"归约论断(Reduction Argument)"和"混合技巧(Hybrid Technique)。"

假设定理结论不成立，则存在 PPT 算法 A 以及多项式 $q(\cdot)$，$p(\cdot)$，使得下式成立：

$$|\Pr[A(X_n^{(1)},\cdots,X_n^{(q)})=1]-\Pr[A(Y_n^{(1)},\cdots,Y_n^{(q)})=1]|>q(n)/m(n)+1/p(n)$$

定义混合随机变量 $H_n^k=(X_n^{(1)},\cdots,X_n^{(k)},Y_n^{(k+1)},\cdots,Y_n^{(q)})$，这里 $0\leqslant k\leqslant q(n)$。显然有 $H_n^0=(Y_n^{(1)},\cdots,Y_n^{(q)})$，$H_n^q=(X_n^{(1)},\cdots,X_n^{(q)})$成立。

设计算法 A'：输入 α(设位于 X_n 或 Y_n 的值域内)

① 随机选择 $k\in\{0,1,\cdots,q(n)-1\}$。

② 生成样本观察值 $x^1,\cdots,x^k,y^{k+2},\cdots,y^q$；($x^1,\cdots,x^k,y^{k+2},\cdots,y^q$ 分别是相应 $X_n^{(i)}$ 或 $Y_n^{(i)}$ 的样本观察值)。

③ 输出 $A(x^1,\cdots,x^k,\alpha,y^{k+2},\cdots,y^q)$。

由全概率公式(注意 k 在$\{0,1,\cdots,q(n)-1\}$上均匀分布)易见有下式成立

$$\begin{aligned}
&|\Pr[A'(X_n)=1]-\Pr[A'(Y_n)=1]|\\
&=1/q(n)\left|\sum_{k=0}^{q-1}(\Pr[A(H_n^{k+1})=1]-\Pr[A(H_n^k)=1])\right|\\
&=1/q(n)\,|\Pr[A(H_n^q)=1]-\Pr[A(H_n^0)=1]|\\
&=|\Pr[A(X_n^{(1)},\cdots,X_n^{(q)})=1]-\Pr[A(Y_n^{(1)},\cdots,Y_n^{(q)})=1]|/q(n)\\
&\geqslant 1/m(n)+1/q(n)p(n)
\end{aligned}$$

这与概率空间 $X=\{X_n\}_{n\in N}$，$Y=\{Y_n\}_{n\in N}$是弱计算不可区分的这一性质矛盾，证毕。

不难看出，定理 3.8 正是引言中所给出的基于口令的安全协议的安全性定义的理论依据。

3. 弱单向函数暗示弱伪随机生成器的存在

单向函数在密码学中处于基础地位，这里给出弱伪随机性与弱单向函数之间的关系。基本结论是，正如单向函数的存在性暗示伪随机生成器的存在性一样，弱单向函数暗示弱伪随机生成器的存在。

定理 3.9 设 G 是扩展因子为 $l(n)=2n$ 的弱伪随机生成器，令 $f:\{0,1\}^*\rightarrow\{0,1\}^*$，定义 $f(x,y)=G(x)$，这里$|x|=|y|$，则 f 是弱单向函数。

证明 显然 f 是易于计算的。下面仍然采用归约论断。假设 f 不是弱单向函数，则对任意多项式 $p(\cdot)$，存在 PPT 算法 A，对足够大的 n，有 $\Pr[A(f(U_{2n}))\notin f^{-1}f(U_{2n})]<1/p(n)$，亦即

$$\Pr[f(A(f(U_{2n})))=f(U_{2n})]>1-1/p(n)$$

注意由 f 的定义有 $f(U_{2n})=G(U_n)$ 成立。

构造算法 D：输入 $\alpha\in\{0,1\}^*$；调用算法 A 求 $\alpha\in\{0,1\}^*$ 在 f 作用下的原像，如果成功则输出 1，否则输出 0。

显然 $D(\alpha)=1\Leftrightarrow f(A(\alpha))=\alpha$。再由 $f(U_{2n})=G(U_n)$，故有

$$\Pr[D(G(U_n))=1]=\Pr[f(A(G(U_n)))=G(U_n)]>1-1/p(n)$$

另外，根据 f 的构造方法，至多有 2^n 个不同的 $2n$ 长比特串在 f 作用下有原像，即 $\Pr[D(U_{2n})=1]=\Pr[f(A(U_{2n}))=U_{2n}]\leqslant 1/2^n$，进而得到 $\Pr[D(G(U_n))=1]-\Pr[D(U_{2n})=1]>1-1/p(n)-1/2^n\geqslant 1/m(n)+1/p(n)$（$n$ 足够大），这与 G 是弱伪随机生成器矛盾。证毕。

推论 3.1 在定理 3.9 的条件下，f 是 $(1/m(n))$ 单向函数。

证明 类似定理 3.9 的证明方法，此处从略。

4. 弱伪随机性理论的密码学应用

如前所述，在考虑像基于口令的安全协议这类问题时，弱伪随机性作为基本安全目标已经足够满足实际应用需求。换句话说，弱伪随机性理论就是基于口令的安全协议的密码学理论基础。

就其他应用而言，还可以考虑把弱伪随机性生成器作为构造密码生成器（如序列密码）的基本组件之一，这样做的优点是显然的，如实现效率可能更高、实现成本更低、选取范围更广。因为以上讨论表明，弱单向函数是弱伪随机生成器的基础，而弱单向函数无疑比单向函数的选取范围要大，条件也弱得多，因此利用弱伪随机生成器构造伪随机生成器是完全可行的。

3.5.2 基于口令的安全协议的模块化设计与分析理论

文献[25]中最早提出了以 BCK 模型为代表的安全协议的模块化设计与分析思想，但其适用范围并不包括基于口令的安全协议。本节主要研究基于口令的安全协议的模块化设计与分析理论，由于基于口令的安全协议的密码学理论基础存在很大的特殊性，即和一般安全协议存在本质区别（如存在字典攻击等），因此这种推广不是平凡的。

1. 基本框架

安全协议的设计难点主要在于安全认证通信问题，通常必须面对这样的主动敌手：该敌手控制联系各合法通信方的公开信道，可以修改、删除甚至是错误地传播消息（伪造）；还能控制消息的延迟，甚至可能还具有额外的能力，如收买某个或某些合法方等。解决安全认证问题的关键就是要确保系统的行为尽可能类似在理想认证信道上通信，传播消息不会被修改或错误提供起源。

文献[25]中把网络安全通信协议视为一种“消息驱动”(Message-Driven)协议。所谓消息驱动协议是这样一个迭代过程,某一具有内部初始状态(协议的输入、随机输入及身份等信息)的通信方可以调用协议;一旦调用,协议即处于等待激活状态,这种激活可能由两类事件引发:来自网络的消息到达或某外部请求(形式化了来自该用户运行的其他进程的信息);一旦激活,协议利用初态及传来的数据,生成一个新的内部状态,同时生成一个发往网络的消息和一个外部请求,此外还产生一个输出值(输出视为累加的)。一旦激活完成,协议等待下一次激活。把各类基于口令的安全协议(诸如基于口令的会话密钥分配协议以及下面要研究的口令更换协议等)都视为消息驱动协议。

安全协议模块化研究方法即 BCK 模型[25]的关键组成部分是构造“通用编辑器 C”(也称认证器):把理想认证模型中的任何安全的安全协议 π 转化成现实非认证模型中的协议 $\pi'=C(\pi)$,后者完成与前者类似的任务,但却能够抵抗能力强得多的现实敌手的攻击。也称这样的协议编辑器为认证器。文献[25]中还进一步指出,认证器的设计可以归结到更简单的 MT 认证器(消息传送认证器)的设计,这是一种结构很简单的协议模块,目标仅限于认证通信双方之间简单的消息交换或传递(Message Transmission),因此设计要简单得多。

在现实模型中设计安全协议的一般方法是:在第一阶段,给出该协议在理想认证通信模型中的可证明安全性设计;在第二阶段,应用协议认证器,把第一阶段的协议“编译”成现实模型中的安全协议。这两个阶段的协议可以相对独立地设计与分析。

认证器的定义涉及对认证模型和非认证模型的严格研究,而且要定义两种模型中所运行协议的“等价性”概念。可证明安全性理论的“模仿论断”有助于定义这一概念:一个协议的行为被另一协议模仿。“模仿”的定义涉及伪随机性理论和计算不可区分概念。

本节旨在研究严格基于口令的安全协议,因此需要基于弱伪随机性理论推广上述“协议模仿”的概念,进而给出基于口令的安全协议的模块化设计与分析理论。

在以下的讨论中,把定义 3.10 中的多项式 $m(n)$ 取为口令字典基数 $|D|$,无妨也称为弱计算不可区分,为方便起见,用“$\overset{wc}{\equiv}$”表示弱计算不可区分。

前面所述的 MT 认证器可以自然推广到基于口令的 MT 认证器,记为 PMT 认证器(基于口令的消息传送认证器)。

(1) 认证模型 AM。不失一般性,只考虑两方通信情形,推广到多方网络情形是自然的。设用户 P_1、P_2 各自运行相同的一个消息驱动协议 π。协议的计算由各方的一个激活序列组成。需要指出,敌手 A(形式化为一个 PPT 算法)可以控制和编排激活,即 A 决定最初激活哪一方以及另一方收到哪一个消息和外部请求等。敌手知道发出的消息、外部请求及协议的输出,但新的内部状态则是未知的。

在理想认证模型中,敌手 A 被限制只能忠实地传递消息(但可以使用延迟、重排等攻击手段)。不失一般性,假设每一个发出的消息都是不同的,即同一消息不会发送两次(在实践中可以这样理解:每一消息都携带发方和收方的身份、时间戳或其他防重放攻击的随机数,因此消息相同的概率至少也是可以忽略的)。当某一方发出一份消息,该消息即被加入“待发送消息”集合 M(形式化记法);无论何时敌手用送达消息 m 激活另一方时,那么必然有

$m \in M$,而且被激活方就是收方(即 m 不可能是敌手伪造的)。敌手还可以随意收买某个或某些用户,这时可以获知被收买用户的所有内部状态,而且从这个时刻起,敌手可以向 M 注入伪造消息。总之在理想认证模型中,敌手是被动的,不具有伪造或篡改消息的能力。

安全协议的整体输出(Global Output)是各方包括敌手的累加输出的连接。敌手的输出是敌手的观察(View)的函数,敌手的观察是指协议执行期间敌手看到或推导出的信息及随机输入等。

下面先引入一些符号。

$ADV_{\pi,A}(x,r)$:当敌手 A 和执行协议 π 的双方交互时,设协议的输入是 $x=x_1x_2$,$r=r_0r_1r_2$(这里 r_0 表示敌手的随机输入,x_i、r_i 表示 P_i 的输入和随机输入);$ADV_{\pi,A}(x,r)$表示 A 的输出。根据具体情况输入可以为空,后不注。

$AUTH_{\pi,A}(x,r)_i$:表示在敌手 A 存在的条件下,P_i 在执行完以上协议之后得到的累加输出。

令 $AUTH_{\pi,A}(x,r)=ADV_{\pi,A}(x,r)AUTH_{\pi,A}(x,r)_1AUTH_{\pi,A}(x,r)_2$,当随机输入 r 均匀随机选取时,用随机变量 $AUTH_{\pi,A}(x)$来表示。

(2) 非认证模型 UM。这是现实模型。UM 模型基本上和 AM 模型类似,除了这时的敌手是一个 UM 敌手(主动敌手)U,U 不再局限于忠实传递 M 中的消息,相反可以使用任何伪造的消息激活任一方。

类似地,可以类比引入符号 $Unauth_{\pi,A}(x,r)$和 $Unauth_{\pi,A}(x)$等,其基本含义类似。

(3) 协议模仿。协议模仿概念是为了表示以下想法:在非认证网络中运行协议 π'与在理想认证模型中运行协议 π 有相同的效果(在功能特别是在安全性能方面是"等价"的)。

下面利用弱计算不可区分概念给出 π'模仿 π 的概念。

定义 3.15 设 π、π'都是 2 方消息驱动协议,称 π'在非认证模型中模仿 π,如果对任一 UM 敌手 U,存在一个 AM 敌手 A,对于所有输入 x,有

$$Unauth_{\pi',U}(x) \stackrel{wc}{\equiv} Auth_{\pi,A}(x)$$

定义 3.15 可以理解为:任何 UM 敌手攻击协议 π'造成的后果和 AM 敌手攻击协议 π 造成的后果是等效的,即前者可以被后者模仿。

定义 3.16(基于口令的安全协议认证器) 对任何协议 π 而言,基于口令的安全协议认证器是这样一个协议编译器 C,$C(\pi)$在非认证网络中模仿 π。

至此,基本思想已经明晰:利用基于口令的协议认证器 C,可以把理想认证模型中安全的基于口令的安全协议转化成现实非认证模型中安全的同类协议。下面考虑认证器 C 的构造。如前所述,C 的设计可以归结到更基本的协议模块设计,即 PMT 认证器。

2. PMT 认证器的设计

设计基于口令的认证器 C 的一般构造技术:首先设计一个以发送消息的外部请求为输入、按认证方式发送消息的"底层"协议 λ;其次,对于任何在理想认证模型中运行的协议 π,认证器输出一个基于口令的协议 π',与 π 几乎完全一样,除了消息传递经过 λ 实现。

更精确的表述:考虑理想认证模型中运行的消息传送协议 MT,对于每次激活,P_1 发送

消息(P_1,P_2,m)给 P_2,输出“P_1 发送 m 给 P_2”,P_2 则输出“P_2 从 P_1 收到了 m”。如果上述基于口令的协议 λ 在非认证模型中模仿(理想认证模型中的)协议 MT,称 λ 是一个基于口令的 MT 认证器,记为 PMT 认证器。

至此可以按以下方式定义协议编译器或认证器 C_λ。对于任一在理想认证模型中运行的协议 π,现实模型中的协议 $\pi'=C_\lambda(\pi)$有以下定义:先调用 λ,对于 π 发出的每一份消息,π'使用“发送该消息给指定方”的外部请求激活 λ;无论何时 π'被某到来的消息激活,立刻用它激活 λ,当 λ 输出“P_i 从 P_j 收到消息 m”,则 π 被来自 P_j 的消息激活。

定理 3.10 设 λ 是一个 PMT 认证器,则以上定义的 C_λ 是一个基于口令的认证器(或编译器)。

证明 设 π 是一个理想认证模型中的协议,只需证明 $\pi'=C_\lambda(\pi)$在非认证模型中模仿 π。即对任意 UM 敌手 U,构造一个 AM 敌手,对于任何输入,满足

$$\mathrm{Unauth}_{\pi',A}(x)\overset{wc}{\equiv}\mathrm{Auth}_{\pi,A}(x)$$

证明方法和文献[34]中的相关部分完全一样,只是将计算不可区分概念替换为弱计算不可区分概念。

下面给出一个基于认证码(MAC 算法)的 PMT 认证器。

基于认证码的 PMT 认证器 λ_{MAC}。

参数设置:发方 A、收方 B 仅仅共享口令 $pw\in D$。待传递消息为 m。其余参数的设置同 3.5.3 小节中的 DH 协议。

协议执行过程如下。

(1) A 随机选择 $N_A\in Z_p^*$,发送($m,C_A=E_{pw}(g^{N_A})$)给 B。

(2) B 收到以上消息后,随机选择 $N_B\in Z_p^*$,发送($m,C_B=E_{pw}(g^{N_B})$)给 A。

(3) A 收到消息后,经解密运算得到 g^{N_B},发送($m,C=\mathrm{MAC}_N(m,B)$)给 B,这里 $N=g^{N_AN_B}$。

B 收到该消息后,验证 MAC 值,决定是否接受,如果接受,输出“B 收到来自 A 的消息 m”。

说明:为了简便,无妨设以上待发送消息中总含有意定接收方的 ID 和防重放攻击的时间戳或随机数据,仍简单记为 $\mathrm{MAC}_{N_B}(m)$;E 是一个安全的对称加密算法(这里均指一一对应的分组密码算法,如 AES,并形式化为一个理想密码模型);MAC 是一个安全的认证码算法:MAC 伪造敌手 F(任意 PPT 算法)具有 Oracle MAC,可以用自己随意选择的任何消息 m 询问 Oracle MAC(当然询问次数是多项式数量级);如果最后敌手 F 能够成功伪造一个未询问过的消息(新消息)的 MAC 值的概率是可忽略的,就称 MAC 算法是安全的,形式化表示为

$$\Pr[k\leftarrow\mathrm{KeyGen},(m,C)\leftarrow F^{\mathrm{MAC}_k};\mathrm{Verify}(m,C)=1]=\mu(n)$$

这里 $\mu(n)$是可忽略函数。

定理 3.11 假设对称加密算法 E 是理想密码模型(Ideal-Cipher Model,ICM),且 MAC 是安全的认证码算法,则 PMT 认证器 λ_{MAC}在非认证模型中模仿(理想认证模型中的)消息

传输协议 MT。

证明 设U是一个与λ_{MAC}交互的 UM 敌手，下面只需构造一个 AM 敌手A满足：至多除了概率$\rho=O(1/|D|)+\mu(n)=O(1/|D|)+\varepsilon(\varepsilon=\mu(n)$是可忽略函数)外，$\mathrm{Auth}_{MT,A}()$、$\mathrm{Unauth}_{\lambda_{MAC},U}()$的统计差异可忽略(统计分布相同)。为了方便，仍然只考虑 2 方情形。需要构造一个满足定义 3.15 的 AM 敌手A。

首先当U激活要发送m给被模仿方P'_2的被模仿方P'_1时，AM 敌手A激活认证模型中的对应方P_1，P_1打算发送m给P_2；A继续运行U和运行λ_{MAC}的被模仿方之间的交互。当被模仿方P'_2输出"P'_2收到来自P'_1的消息m"时，A使用来自P_1的消息m激活认证模型中的P_2。当现实敌手U收买某一方时，A收买理想认证模型中对应的一方。最后A输出U输出的任何信息。

设 Bad 表示以下"模仿失败"事件：P'_2输出"P'_2收到来自P'_1的消息m"，同时P'_1未被收买，而(m,P_1,P_2)当前不在待传递消息集合M当中(即P'_1事实上没有发送过以上的消息)。显然，如果模仿不失败，AM 敌手和 UM 敌手的输出统计分布是完全相同的，因此下面只需证明 Bad 发生的概率 $\Pr[\mathrm{Bad}]\leqslant\rho$。

在 Bad 发生的情况下，这意味着U最终成功伪造了一个新的 MAC 值，其基本思路是：在理想密码模型中，E_{pw}是一个随机置换，因此敌手不可能以大于ρ的概率得到(pw,N_B)的任何信息，除了在线穷举pw。再由 MAC 是安全的，因此 Bad 发生的概率至多是ρ。

首先要注意，假设对称加密算法E_{pw}是随机预言，即E_{pw}是一个一一对应的随机函数。因此询问者即敌手U在不经询问 Oracle E_{pw}的情况下，也可能导致 Bad 事件的发生，但由于这是随机预言，再考虑到口令$pw\in D$，因此这时(伪造任何一个新消息的对应 MAC 值事件)发生概率至多是$O(1/|D|)$(模仿了在线口令猜测攻击)。因此，以下只需考虑除了这种模仿失败情况之外，即在必须询问随机预言的情况下，事件 Bad 的发生概率ε是否是可忽略的。

下面假设不然，即这种情况下 Bad 发生的概率至少是ε，并且是不可忽略的。不失一般性，假设发生概率就是ε。下面以之为子程序构造一个 MAC 伪造者F。为了方便，仍然记现实模型中的通信双方为A、B，敌手仍然记为U。

在事件 Bad 以概率ε发生的条件下，设l是现实敌手U在运行协议过程中传递的消息总数，也就是说，U至多l次使用"消息……"激活各方。那么在至多l次询问相应 MAC Oracle 的条件下，MAC 伪造者F的成功概率就是ε/l。

更精确的描述：可以定义一个 MAC 伪造者F，其输入是N^*，$C_A=E_{pw}(g^{N_A})$，$C_B=E_{pw}(g^{N_B})$，这里$N^*=g^{N_AN_B}$(N^*显然是未知的)。F拥有随机预言E_{pw}和以N^*(未知)为验证密钥的 MAC Oracle MAC_{N^*}。

MAC 伪造者F的构造：F将模仿用户A、B与敌手U交互运行协议(提供模拟环境，敌手可以从中得到"观察"即 View)。设m^*是用以激活B的全部消息中随机选定的一个消息。

如果模仿过程中A被收买，则F宣告失败，放弃模仿。

如果被模仿方A被密文$c=E_{pw}(g^N)$激活(这里设 $\mathrm{List}=\{(N,c)\mid c=E_{pw}(g^N)\}$是$F$保

存的一份记录,开始是空集;设 c 是对接收到的消息 $m \neq m^*$ 的回答),则 F 首先从 List 中检索是否有对应于索引 c 的询问,如果有则选取对应的 N;否则,随机选取 $N \in Z_p^*$,定义 $c = E_{pw}(g^N)$,依据 DDH 假设,随机选取 k,并用 $\text{MAC}_k(g^N)$ 予以回答;最后 F 刷新 List 表(把新的(N,c)储存进 List)。如果 $c=C_A$ 或 C_B,则放弃。

当 B 被敌手用消息 m^*,C_A 激活,则 F 用(m^*,C_B)回答,注意 C_B 恰好就是 F 的输入密文。

如果 A 被来自 B 的消息"(m,C_B)"激活,且 $m \neq m^*$,则 F 随机选取 N,回答 $c = \text{MAC}_N(m)$;如果 C_A 曾经询问过,则 F 询问 MAC Oracle MAC_{N^*} 作为回答;而如果 $m = m^*$,F 放弃模仿,宣告失败。

最后,如果敌手 U 使用来自 A 的消息"(m^*,C)"激活 B,则 F 输出(m^*,C)(期望 $C = C^* = \text{MAC}_{N^*}(m^*)$),停机。

对以上算法可做以下分析。

首先,在 F 不放弃模仿的情况下,敌手 U 在和 F 的模仿交互过程中得到的观察与和现实用户交互得到的观察的统计分布相同。然后设 Bad^* 表示以下事件:在 F 的模仿运行中,用户 A 被假冒,且对应的假冒消息是 m^*。

由于 m^* 是随机选取的,而且 Bad 事件和放弃模仿事件不会同时发生,因此有事件 Bad^* 的发生概率为 $\Pr[\text{Bad}^*]=\varepsilon/l$。

下面假设 Bad^* 发生,这时 m^* 就是敌手的假冒消息;这时 B 最后接收到的消息必然是一个合法的 MAC 值(对应验证密钥正是 N^*),即

$$C = C^* = \text{MAC}_{N^*}(m^*)$$

而且 A 从未生成过对应消息的这个 MAC 值(注意 A 从未被"发送消息 m^* 给 B"的外部请求激活过,当然可以预先已经假定任何一方不会两次发送同一消息,这在技术上很容易做到,如附加时间戳等;另外,任何两个消息对应同一密钥 N^* 的概率是$1/2^k$)。总之可以看出,F 从未用消息 m^* 询问过 MAC Oracle MAC_{N^*},因此

$$C = C^* = \text{MAC}_{N^*}(m^*)$$

是一个成功的伪造,即 F 以概率 ε/l 成功伪造了一个新消息的 MAC 值,如果 ε 是不可忽略函数,必然 ε/l 也是不可忽略概率,这与 MAC 算法是安全的前提条件矛盾。因此 $\varepsilon=\mu(n)$ 是可忽略的。

综上所述可以看到,事件 Bad 至多以概率 $\rho=O(1/|D|)+\mu(n)$发生,这里 $\mu(n)$是可忽略函数。

文献[25]中还指出:对于较复杂协议,可以直接以安全的会话密钥分配协议为 MT 认证器,即每次发送消息时,首先通信双方调用安全的会话密钥分配协议得到会话密钥 K,然后对所有要传送的消息 m 采取$(m,\text{MAC}_K(m))$传送方式。显然,这可以自然推广到基于口令的安全协议,这时 PSKD 协议(基于口令的会话密钥分配协议)就是一个 PMT 认证器。

3. 设计基于口令的安全协议的一般方法

至此可以利用推广后的安全协议模块化设计与分析理论研究各类基于口令的安全协

议，其一般设计方法如下。

(1) 设计 PMT 认证器(如上述给出的 λ_{MAC})。

(2) 设计在理想认证模型中可证明安全的基于口令的安全协议 P_{ideal}。

(3) 根据 PMT 认证器给出基于口令的协议认证器(编译器)C_λ，即任何消息传递都通过 PMT 实现。

(4) 使用 C_λ 编译得到现实模型中可证明安全的基于口令的安全协议 P。

利用这种方法，有助于实现基于口令的安全协议设计与分析的系统工程化，也有助于在设计阶段即可排除可能的问题和冗余步骤。

3.5.3　基于口令的会话密钥分配协议

下面将利用 PMT 认证器 λ_{MAC} 为基本模块构造基于口令的会话密钥分配协议 λ_{pw}。

1. 理想认证模型中可证明安全的会话密钥分配协议 $\lambda_{ideal\text{-}pw}$

无妨记基于口令的会话密钥分配协议为 PSKD 协议。由于理想认证模型中只存在被动敌手，因此协议的安全性就是指会话密钥的机密性。首先给出理想认证模型中安全的 PSKD 协议 $\lambda_{ideal\text{-}pw}$，即著名的 Diffie-Hellman 协议，简称为 DH 协议。

1) DH 协议

参数设置：公开素数 p、q，满足 $q|(p-1)$，g 是有限域 F_p 的一个 q 阶生成元；通信双方 A(发起方)、B 共享秘密口令字 pw(由于是在理想认证模型中，实际上并不需要 pw)。

协议执行：发起方 A 随机选取 $x\in_R Z_p^*$，发送消息 $g^x \bmod p$ 给收方 B；接到消息后，B 随机选择 $y\in_R Z_p^*$，发送消息 $g^y \bmod p$ 给 A。

DH 协议的执行结果：通信双方 A、B 达成共享会话密钥 $K=K_A=(g^y)^x=K_B=(g^x)^y=g^{xy} \bmod p$。

定理 3.12[25]　在 CDH(计算 Diffie-Hellman)问题难解的意义下，以上的 DH 协议在理想认证模型中是安全的。

2) 对 DH 协议的进一步修改——同时生成多重会话密钥

为了更好地确保共享会话密钥 $K=K_A=K_B$ 的品质，隐藏密钥数据的统计结构，引入安全 Hash 函数 $H:\{0,1\}^\infty\to\{0,1\}^n$，即在不增加协议交互步骤的前提下，把共享会话密钥修改为：$K=H(g^{xy}||A||B)$。由于 H 是伪随机函数，因此密钥数据的统计结构被有效“遮蔽”，在 RO 模型中，这样的协议当然还是安全的。

文献[38]中指出，在网络通信中有时要根据需要同时生成多重会话密钥；基本的要求就是实现代价要尽可能低，同时抵抗已知会话密钥攻击。

基于这种考虑，协议可进一步修改为：仍然不增加交互步骤，生成 m 重共享会话密钥

$$K_1=H(g^{xy}\parallel A\parallel B),K_2=H(g^{xy}\parallel K_1),\cdots,K_m=H(g^{xy}\parallel K_{m-1})$$

这里 m 自然应该限制在多项式数量级。

在 RO 模型中，上述修改协议仍然是安全的。在 RO 模型中 Hash 函数视为随机函数，即 Hash 函数的输出并不依赖于输入，因此即使已经暴露任意$(m-1)$个会话密钥，无妨假设

就是 $K_1,\cdots,K_{m-1}$，基于 CDH 假设 g^{xy} 仍然未知；敌手只能做多项式次 Oracle 询问，因此根据 RO 模型方法论，任何求解 K_m 的 PPT 算法的成功概率仍然是可忽略的。因此得到以下推论。

推论 3.2 在 CDH 问题难解的意义下，如上修改后的 DH 协议，在理想认证模型中是安全的。

2. 把修改后的 DH 协议“编译”成现实模型中的 PSKD 协议 λ_{pw}

利用 PMT 认证器 λ_{MAC} 构造协议编译器或认证器 C_λ，就得到现实模型中安全的 PSKD 协议 λ_{pw}。

参数生成是类似的，协议执行过程如下。

(1) A 随机选取 $x\in Z_p^*$，发送 $E_{pw}(g^x)$ 给收方 B。

(2) B 随机选取 $E_{pw}(g^y)$，$y\in Z_p^*$，发送 $E_{pw}(g^y)$ 给 A。

(3) A 收到消息后，经解密运算得到 g^y，发送 $C_A=\text{MAC}_N(A,B,g^x)$ 给 B，这里 $N=g^{xy}$。

(4) B 发送 $C_B=\text{MAC}_N(B,A,g^y)$ 给 A。

协议执行结果：A、B 达成多重共享会话密钥。

$$K_1=H(g^{xy}\parallel A\parallel B),K_2=H(g^{xy}\parallel K_1\parallel A\parallel B),\cdots,K_m=H(g^{xy}\parallel K_{m-1}\parallel A\parallel B)$$

定理 3.13 在以上条件下，λ_{pw} 是安全的 PSKD。

证明 由定理 3.11，显然成立。

3. PSKD 协议 λ_{pw} 的其他性质分析

从上述 PSKD 协议的构造原理易于看出，除了实现效率较为理想之外，PSKD 协议 λ_{pw} 还满足前向安全性(Forward Security)：即使某次通信结束后，秘密口令字 pw 泄漏；这时，当前时刻的会话密钥以及之前的所有会话密钥仍然是安全的。原因在于会话密钥并不直接依赖于口令字。前向安全性在降低系统安全风险方面发挥着重要作用，如下文要介绍的口令更换协议。

另外前面还指出，对于较复杂的基于口令的安全协议，直接以 λ_{MAC} 为基本模块构造协议认证器还是比较麻烦。如前所述，事实上协议编译器 C_λ 还可以直接用会话密钥分配协议 λ_{pw} 为 PMT 认证器构造：这时所有的消息传送都要先经过调用 λ_{pw}，然后再以

$$(m_i,\text{MAC}_{K_i}(m_i))\quad i=1,2,\cdots,m$$

的形式发送消息。

可以看出，由于具有同时生成多重会话密钥的性质，因此只需调用一次协议就足以满足复杂协议多次收、发消息的需求。

3.5.4 口令更换协议的模块化设计与分析

设计 PPC 协议的难点之一在于，必须兼顾考虑对原有口令、待更换口令的“双重字典攻击”问题。文献[34]中协议的缺陷主要有两点：没有考虑到“双重字典攻击”问题；认证措施不完备。本节考虑利用协议的模块化设计与分析观点研究 PPC 协议，即提出 PPC-New 协议。

1. 理想认证模型中安全的 PPC 协议——PPC_{ideal}

在理想认证模型中要安全传送由 A 产生的新口令是很容易的：只需利用某安全对称加密算法 E 加密传送 $pw'\in D$ 即可：设调用 PSKD 协议产生的会话密钥为 $K_1=H(g^{xy}\|A\|B)$，只需加密传送消息 $m=E_{K_1}(pw')$给收方 B 即可。

PPC_{ideal} 如下：

(1) A、B 调用基于口令的会话密钥分配协议，生成共享会话密钥 $K=H(g^{xy}\|A\|B)$。

(2) A 传送消息 $m=E_K(pw')$ 给 B。

(3) B 解密就得到更换后的口令 pw'。

定理 3.14 如果对称加密算法 E 是安全的，则在理想认证模型中 λ_{pw} 是安全的。

证明 在理想密码模型中上述结论是显然成立的。

事实上很容易看出，类似文献[36]中的双重字典攻击不再成立，因为根据定理条件，敌手不可能得到任何新、旧口令的关联等式。

2. 现实模型中的口令更换协议 PPC-New

PPC-New 如下：

(1) A、B 调用基于口令的会话密钥分配协议 λ_{pw}，生成共享会话密钥 $K_1=H(g^{xy}\|A\|B)$，$K_2=H(g^{xy}\|K_1\|A\|B)$。

(2) A 随机选取 $pw'\in D$，传送消息 $m=E_{K_1}(pw')$给收方 B。

(3) B 随机选取 $N_B\in\{0,1\}^k$，发送$(m,E_{K_2}(N_B))$给 A。

(4) A 传送$(m,MAC_{N_B}(m))$给 B。

B 的验证、接收过程从略。

显然这是利用 PMT 认证器 λ_{MAC} 作为基本模块得到的现实模型中的一种消息传送协议。还可以直接以会话密钥分配协议 λ_{pw} 为 PMT 认证器设计传输协议：设执行 λ_{pw} 后双方得到 2 重会话密钥 K_1，K_2；A 随机选取 $pw'\in D$，传送消息$(m=E_{K_1}(pw'),MAC_{K_2}(m))$给收方 B；B 验证认证码，如果接受，则 A、B 双方已经成功更换了新的口令 pw'。

这样协议就得到了进一步简化。

根据定理 3.12、定理 3.13 及推论 3.2 等易于看出，口令更换协议 PPC-New 是可证明安全的。

上面讨论了基于口令的安全协议的密码学理论基础，特别是通过推广文献[36]中的观点，提出了研究这类协议模块化构造理论，并具体给出了一个基于认证码算法的 PMT 认证器，这是构造协议编译器的基本模块。实践表明，这种方法具有一般意义。

利用上述理论工具还成功设计了两类基于 Diffie-Hellman 问题的安全协议：PSKD 协议和 PPC-New 协议。就 PSKD 协议而言，除实现效率较高之外，还具有同时生成多重会话密钥的功能，这种性质最初由文献[38]中提出，这是一种非常有用的性质。文献[38]中的协议也是基于 Diffie-Hellman 问题的，但需要每方一次产生 n 个随机数，才能生成$(n-1)^2$ 个会话密钥；与之相比，设计的 PSKD 协议生成多重会话密钥的效率更高：每方只需一次生成一个随机数，就可以安全生成满足任何实际需要的多重会话密钥。此外，文献[38]中的协议

并未给出安全性证明，而在理想密码模型中给出了以上协议的安全性证明。PPC-New 协议和现有结果[34]比较：尽管都是基于 Diffie-Hellman 问题的，但由于采用了模块化设计与分析的观点，协议步骤简洁明了，特别是还具有可证明安全性；而 PPC 协议[34]却不能抵抗"双重字典攻击"和拒绝服务攻击。

另外，由 PSKD 协议 λ_{pw} 具有前向安全性，易于推导出 PPC-New 协议也满足前向安全性。其应用含义是：设某一时刻通信结束后，原有口令字 pw 泄露，则本次通信所更换的新口令 pw′仍然是安全的。也就是说，旧的口令泄露不会造成任何安全损失。

3.6 小结

本章试图全面、系统地介绍可证明安全性理论与方法的发展历程、主要研究方向、有代表性的方法论和重要结果，以及作者的一些研究成果和观点。作者认为读者要想真正理解和掌握这些研究成果，首先应精读文献[1]，并仔细阅读本章所列其余参考文献。本章在叙述上尽可能采用模型化观点，以有助于说明方法论。关于可证明安全性理论与方法的研究综述可参阅文献[39]，关于信息安全中的计算复杂性理论基础知识可参阅文献[40]。

参考文献

[1] Goldreich O. 密码学基础. 北京：电子工业出版社，2003.

[2] Goldwasser S，Micali S. Probabilistic Encryption，Journal of Computer and System Science，28:270～299，1984.

[3] Goldwasser S，Micali S，Rivest R. A Digital Signature Scheme Secure Against Adaptive Chosen-Messege Attacks. SIAM Journal of Computing，17(2):281～308，April 1988.

[4] Mihir Bellare，Phillip Rogaway. Random Oracles are practical：A Paradigm for Designing Efficient Protocols，in the Proceedings of the First ACM Conference on Computer and Communicatuions Security，ACM，November 1993.

[5] Ran Canetti，Oded Goldreich and Shai Halevi，the Random Oracle Methodology，Revisited. http://www.wisdom.weizman.ac.il/~Oded/cryptography.html，1998.

[6] David Pointcheval. Asymmetric Cryptography and Practical Security，Journal of Telecommunication Technology，Volume 4/2002，41～56.

[7] Fiat A，Shamir A. "How to prove yourself：practical solutions to identification and signature problems，"Advances in Cryptology -crypto'86 Proceedings，Lecture Notes in Computer Science Vol. 263，A. Odlyzko ed.，Springer-Verlag，1986.

[8] Mihir Bellare，Bilian J，Phillip Rogaway. "The Security of Cipher block Chaining" Advances in Cryptology-Crypto94 Proceedingss，Lecture Notes in Computer Science Vol. 839，Y. Desmedt ed.，Springer-Verlag，1994.

[9] Mihir Bellare，Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis，editors，Advances in Cryptology-EUROCRYPT'94，Vol. 950 of Lecture Notes in Computer Sciences，92～111. Springer-Verlag，1995，9～12 May，1994.

[10] Mihir Bellare, Phillip Rogaway. The Exact Security of Digital Signatures-How to Sign with RSA and Rabin, Advances in Cryptology -Eurocrypt 96 Proceedings, Lecture Notes in Computer Science Vol. 1070, U. Maurer ed. , Springer-Verlag, 1996.

[11] Silvio Micali, Leonid Reyzin. Improving The Security Of Digital Signature Schemes, http://theory.lcs.mit.edu/~ reyzin, August 25, 1999.

[12] Pointcheval D, Stern J. Security proofs for signature schemes. In Advances in Cryptology-EUROCRYPT'96, volume 1070 of Lecture Notes in Computer Science, 387～398, Springer-Verlag, 1996.

[13] Mihir Bellare, Gregory Neven. Transitive Signatures based on Factoring and RSA. In Advances in Cryptology-ASIACRYPT'02, Lecture Notes in Computer Science Vol. , Y. Zheng ed. , Springer-Verlag, 2002.

[14] Eu-Jin Goh, Stanislaw Jarecki. A Signature Scheme as Secure as the Diffie-Hellman Problem, Advances in Cryptology- EUROCRYPT'03, Lecture Notes in Computer Science, Springer-Verlag, May 2003.

[15] Francois Koeune. Careful design and intergration of cryptographic primitives with contributions to timing attack, padding schemes and random number generators. These soutenue en vue de l'obtention du grade de Docteur en Sciences Appliquees. Louvain-la-Neuve, Belgique, Juillet, 2001.

[16] Rosario Gennaro, Shai Halevi, Tal Rabin. Secure Hash-and -Sign Signatures without the Random Oracle. March 22, 1999.

[17] Cramer R, Shoup V. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack, Advances in Cryptology-Crypto 1998, volume 1462 of Lecture Notes in Computer Science, 13～25, 1998.

[18] Needham R, Schroeder M. "Using encryption for authentication in large networks of computers," Communications of the ACM, Vol. 21, No. 12, 993～999, December 1978.

[19] Sacco G. "Timestamps in key distribution protocols," Communications of the ACM, Vol. 24, No. 8, 523～536, 1981.

[20] Burrows M, Abadi M, Needham R. "A Logic for authentication," ACM Transactions on Computer systems, Vol. 8, No. 1. 1990.

[21] Bellare M, Rogaway P. "Entity authentication and Key Exchange", Advances in Cryptology-Crypto'93 Proceedings, Lecture Notes in Computer science Vol. 773, D. Stinson ed. , Springer-Verlag, 1993.

[22] Mihir Bellare. Provably Secure Session Key distribution—The Three Party case. In Proceedings of the ACM Symposium on the Theory of Computing, May, 1995.

[23] Mihir Bellare. The Challenge of Session-Key Distribution Protocols, university of Calufornia at San Diego, 2000.

[24] Bellare M, Pointcheval D, Rogaway P. Authenticated Key Exchange Secure against Dictionary attacks. In EuroCrypt 2000, Springer-Verlag(LNCS 1807), 139～155, 2000.

[25] Mihir Bellare, Ran Canetti, Hugo Krawczyk. A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols. In Proceedings of the 30th Annual Symposium on the Theory of Computing, ACM, 1998.

[26] Micali S, Rogaway P. "Secure Computation", Manuscript, 1992. Preliminary version in Advances in Cryptology-Crypto'91 Proceedings, Lecture Notes in Computer Science Vol. 576, Feigenbaum J ed., Springer-Verlag, 1991.

[27] Bellovin S M, Merritt M. Encrypted key exchange: Password-Based protocols secure against dictionary attacks. In ACM/IEEE Symposium on Research in Security and Privacy, 72～84, 1992.

[28] Halevi S, Krawczyk H. Public-Key Cryptography and Password Protocols. In 5th ACM Conference on Computer and Communications Security, 122～131, 1998.

[29] Bellare M, Pointcheval D, Rogway P. Authenticated Key Exchange Secure against Dictionary attacks. In EuroCrypt 2000, Springer-Verlag(LNCS 1807), pages 139～155, 2000.

[30] Boyko V, MacKenzie P, Patel S. Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman. In: B. Preneel ed. Advances in Cryptology-EuroCrypt'00 Proceeding, LNCS 1807, Berlin: Springer-Verlag, 2000. 156～171.

[31] Katz J, Ostrovsky R, Yung M. Efficient password-authenticated key exchange using human-memorable passwords. Advances inCryptology-proceedings of EUROCRYPT 2001, Lecture Notes in Computer Science volume 2045, Springer-Verlag, 475～494, 2001.

[32] Oded Goldreich, Yehuda Lindell. Session-key Generation using Human Passwords Only. http://www.wisdom.weizman.ac.il/~oded/cryptography.html. December 3, 2002.

[33] Mario Di Raimondo, Gennaro R. Provably secure threshold password-authenticated key exchange. Advances inCryptology-proceedings of EUROCRYPT 2003, Lecture Notes in Computer Science volume, Springer-Verlag, 2003.

[34] Chang T Y, Yang W P, Hwang M S. Simple authenticated key agreement and protected password change protocol. An international journal computers&mathematics with applications, May 2005, Vol. 49: 703～714.

[35] Yeh H T, Sun H M. Simple authenticated key agreement protocol resist to password guessing attacks. ACM SIGOPS Operating Systems review, 2002, 36(4): 14～22.

[36] Chi-I Wang, Chun-I Fan, Guan D J. Cryptanalysis on Chang-Yang- Hwang protected password change protocol. Cryptology eprint Archive: Report 2005/182. http://eprint.iacr.org/2005/182.

[37] 冯登国，陈伟东. 基于口令的安全协议的模块化设计与分析，中国科学 E 辑，2007，37(2): 223～237. 又见 Dengguo Feng, Weidong Chen: Modular approach to the design and analysis of password-based security protocols. Science in China Series F: Information Sciences 50(3): 381～398 (2007).

[38] Euw-Yi Yang, Jinn-Ke Jan. An Enhanced and secure protocol for authenticated key exchange. Cryptology eprint Archive: Report 2004/182. http://eprint.iacr.org/2004/182.

[39] Feng Dengguo. Research on Theory and Approach of Provable Security, Journal of Software, 16(10) 1743～1756, 2005.

[40] 冯登国. 信息安全中的数学方法与技术. 北京：清华大学出版社，2009.

第 4 章　形式化分析理论与方法

Needham 和 Schroeder[1,2] 首次提出了使用形式化方法对协议进行分析的思想，第 1 篇系统地对协议进行分析的论文是 Dolev 和 Yao 于 1981 年发表的论文[3]，这篇论文具有标志性意义。他们用算法的手段分析了两类特殊协议的安全性。随后，Dolev、Even 和 Karp[4] 用多项式时间算法对某些特定类型协议的安全性进行判定，并且发现这种方法只适用于特殊的协议，对一般的协议是不可判定的[5]。

Dolev-Yao 的工作开启了用形式化方法分析安全协议的先河，为随后进行的工作奠定了基础。他们定义了协议并发运行环境的形式模型，密码算法无关的验证，以及敌手的攻击行为等，形成了后来的所谓 Dolev-Yao 模型。

最早用于安全分析的是由 Millen 开发的 Interrogator 系统[6]，它的基本原理是企图通过遍历整个状态空间，找到协议安全的漏洞。Kemmerer[7,8] 通过传统形式规范语言描述密码协议，与 Millen 相同的是，他将协议规定为状态机，但是他的系统中附着一个证明器，可以把一些性质描述为系统的不变量，证明它们是否被系统保持不变。

在安全协议分析领域，如果把 Dolev-Yao 的工作视为形式化方法的标志性工作的话，Burrows、Abadi 和 Needham 的工作就应该是形式化方法的代表作和里程碑。他们利用知识和信念逻辑描述和推理认证协议。这个逻辑系统被称为 BAN 逻辑(取作者们名字的第一个字母)。当时这是一个全新的方法，用一集公式表示协议主体的信念，或者知识，用一集推理规则从原有的公式得到新的信念公式。这个逻辑可以指出一些协议的漏洞。BAN 逻辑的出现大大激发了人们应用形式化手段分析安全协议的兴趣。沿着这个方法，许多逻辑被构造了出来，其中大多是 BAN 逻辑的变种，目的是弥补 BAN 逻辑的不足，如 GNY 逻辑、AT 逻辑及 SVO 逻辑等。大多是基于模态逻辑，模态逻辑的一个特点就是在多数情况下是可判定的[9]。然而，逻辑推理的方法有着自己的不足。主要表现为逻辑的抽象性较高，这种抽象性往往会掩盖(或丢掉)协议执行的状态信息，因而难以完全反映协议运行的全貌，有着自身的局限性。

在 BAN 之后，许多已有的形式化方法逐渐被用于对安全协议的分析上。这有两方面的原因：其一是受 BAN 逻辑的影响和启发，安全协议的研究者寻找安全协议适当的验证工具，包括量身定做的和对已经存在方法的改制；其二是为许多成熟的形式化方法寻找有意义的应用领域。因此，自 20 世纪 90 年代起，安全协议的形式化分析研究出现了空前的繁荣景象。其中值得一提的是 Gavin Lowe 的工作[10]，在这项工作中，Lowe 成功地应用模型检测的工具 FDR 发现了 Needham-Schroeder 协议的一个漏洞。这个结果对应用模型检测的方法分析安全协议有着典型意义。FDR 是应用进程代数-通信序列进程(CSP)原理开发的模型检测工具，其目的是检测一个通信系统中进程并发运行的性质。对于安全协议的成功应用，激发了模型检测技术领域的研究热情。不仅如此，这个漏洞的发现使得人们更加清醒地认识到安全协议分析的复杂性，因而系统化的分析方法显得愈发重要。

在这两类方法之后，较为新颖的方法就是类型检测(Type Checking)方法[11,12]，这个方

法利用 Spi 演算系统[13]，给每个通道和消息赋予类型，将协议的安全性目标的达到归结为协议执行过程中类型的保持。类型的破坏将导致安全性的丧失。

还有一类基于定理证明的方法，这种方法的主要目的是证明协议的正确性。对于不正确的协议往往缺乏提供有用参考的能力。最具代表性的有：Paulson 的归纳证明法[14]；Thayer Fábrega、Hertzog 和 Guttman 等提出的串空间模型（Strand Space）[15]。定理证明的方法可以通过定理证明器协助完成证明过程。串空间模型是用图的形式表达协议的执行过程。协议的一个丛就是协议的一个并发运行。协议的安全性质通过所有丛保持的性质来刻画。利用串空间原理，Song 发展了一种自动检测工具 Athena[16]。

值得指出的是，Blanchet[17]提供了使用进程演算来描述协议并通过求解器进行自动验证的方法。它在 pi 演算中加入了一些密码原语，该进程演算将事件扩充进来，用以描述对应性。这些事件只是用来表示协议，而不用于证明对应性。然后协议被自动转换为一组 Horn 子句。最后，将这些子句传给一个基于归结（Resolution-Based）的求解器。该求解器并非总是终止的，但它对一种经过标记的协议来说是终止的。实验结果表明，它对很多协议都是终止的。此方法的优点是：它是完全自动的，用户只需对协议和要证的对应性进行编码即可；它不限制协议会话的数目及敌手可处理的项的规模；它可处理很通用的密码原语，包括共享密钥加密、公钥加密、签名、单向 Hash 函数及 Diffie-Hellman 密钥协商；它有明确的语义基础。此方法的缺点是：个别时候算法不终止，同时该技术是不完备的，向 Horn 子句的转换引入了一种抽象，该抽象无法记住每个行为副本的编号。这种抽象对于处理无穷会话来说是关键的，也正是这种抽象提供了证明对应性的充分条件，但对正确协议可能会失败。基本上，在证明那些开始时需要对一些值保密，之后又需要公开这些值的协议时会失败。在实践中，该工具仍然是很精确的，在证明那些正确的协议时它总是成功的。

在基于 Dolev-Yao 模型的安全协议形式化分析方法蓬勃发展时期，另外一种基于计算复杂性的安全性证明方法也悄悄兴起。后一种方法就是在第 3 章介绍的可证明安全性方法。普遍的认识是：基于计算复杂性的方法是（密码学）可靠的。而（大多数）建立于 Dolev-Yao 模型之上的形式化方法没有真正建立起密码学的可靠性。因此，将这两种方法统一到一个框架之中，建立形式化方法的可靠性弥补其不足之处就成为当务之急。近年来，人们的注意力集中到突破 Dolev-Yao 模型之上，以及证明形式化方法的密码学可靠性的结果上。也就是两种方法的融合，人们也把这种方法称为混合方法。

Abadi 和 Rogaway[18,19]定义了加密表达式的简单语言，证明了如果两个表达式模型逻辑公式等价，则它们在计算的解释下，根据计算不可区分的标准概念是等价的。Micciancio 和 Warinschi[20]做了进一步研究，证明了如果使用充分强的加密方案，任何两个表达式计算等价当且仅当它们可以在逻辑下等价。Gligor 和 Horvitz[21]精确刻画了这种等价成立对于加密方案的要求。Micciancio 和 Warinschi[22]进一步给出具有主动敌手的安全协议安全性证明的方法，这是第一个将具有主动敌手的简洁的逻辑转换成标准计算情形下的结果。

值得注意的是 Impagliazzo 和 Kapron[23]的结果，这个结果也是沿着 Abadi 和 Rogaway 的路线，他们给出了两个逻辑系统推理关于标准密码安全定义可靠的密码构造。论文的一个新颖之处是，某些证明利用了模型论非标准分析手段。

Backes、Pfitzmann 和 Waidner 等人[24]给出了一个密码可靠的所谓的“密码库”（cypto-library）：证明了由 Dolev-Yao 模型的要素构成的抽象库，在任意主动敌手存在的情况下，任

意协议环境下对于任意安全性质,可以密码实现。这个库包括公钥加密、数字签名、新鲜值、列表操作和应用数据,主要来源于任意密码安全的公钥加密、签名系统,通过附加的诸如标签和随机化的操作增强其功能。主体的行为环境是交互式的,也有主体的模型、交互模型及敌手模型。其安全刻画也是利用模拟的思想,计算不可区分的机制实现。

正如前面所述,参加协议运行的主体有两类:一类是诚实主体,在协议中的行为一直按照协议的规范进行,如果收到不合规范的消息,则认为与自己无关;另一类主体就是所谓的敌手,或者称为入侵者。对于多数协议来说,可以认为只有一个敌手。他的能力的强弱直接反映到我们使用的敌手模型之中。

形式化分析方法中最为流行的敌手模型是 DY 模型[3]。在协议运行的开始或者运行过程中,敌手可以窃听、消去以及任意安排公开通信通道上的消息。他也可以从观察到的消息产生新的消息,并将其加入到信道之中。敌手可以将非加密消息分成若干个新的消息或者将若干个已知的消息合并生成一个新的消息。敌手可以用自己已知的密钥对任意的消息加密。敌手还可以解密一个收到的密文,前提是敌手知道正确的密钥。敌手可以根据需要,截断信道上传输的任何消息,注入自己产生的新的消息,发送该消息至依赖的目标主体或者任何其他主体。

形式化分析方法的通常做法是:把协议视为状态迁移系统,分析所有可能的轨迹(Trace)。判定安全性质是否在所有的轨迹上被保持。在众多的以轨迹为主的模型中,应用最广的有 CSP[25,26]、高阶逻辑[14]、多集重写系统[27,28]及串空间[15]等。

由于篇幅的限制,为了使得读者理解典型的技术特点,本章选择了一些影响较大的系统进行技术分析,主要包括 BAN 逻辑[1,2]、Kailar 逻辑[29]、Paulson 的定理证明系统[14]、Blanchet 的自动验证系统。同时,本章也介绍了形式化方法的计算可靠性(Soundness,有的学者也翻译成合理性)方面的研究成果。希望能够通过分析,掌握基本的分析思想和手段,为读者进一步研究提供必要的基础。

4.1 BAN 逻辑

BAN 逻辑是用来表达并证明安全协议中信任关系的一种认证逻辑,由 Burrows、Abadi 和 Needham 3 人[1,2]于 20 世纪 80 年代末 90 年代初提出。它将逻辑方法引入到安全协议验证中,极大地激发了一批研究者投入到安全协议的形式化分析研究中,成为安全协议分析的一个里程碑。

4.1.1 基本术语

在 BAN 逻辑中有主体、密钥、语句等几种对象。一般用符号 A、B、S 表示特定主体;K_{ab}、K_{as}、K_{bs}表示特定的共享密钥;K_a、K_b、K_s 表示特定的公钥,而 K_a^{-1}、K_b^{-1}、K_s^{-1} 表示相应的私钥;N_a、N_b、N_c 表示特定的语句。符号 P、Q、R 表示主体变量;X、Y 表示语句变量;K 为密钥变量。

以下是 BAN 逻辑中所用到的一些术语:

(1) P believes X:主体 P 认为 X 为真。

(2) P sees X:P 看到 X。

(3) P said X：P 曾说过 X。

(4) P controls X：P 对 X 具有控制权。

(5) fresh X：公式 X 是新鲜的。

(6) $P\overset{K}{\leftrightarrow}Q$：$P$ 和 Q 共享密钥 K。

(7) $\overset{K}{\mapsto}P$：K 是 P 的公钥。

(8) $P\overset{X}{\rightleftharpoons}Q$：表示 X 是一个只有 P 和 Q 知道的秘密。

(9) $\{X\}_K$：表示用密钥 K 对公式 X 的加密。

(10) $<X>_Y$：表示 X 与 Y 的组合，其中 Y 为一秘密。

4.1.2 逻辑规则

BAN 逻辑的规则主要有以下几组。

(1) 消息含义规则

$$\frac{P \text{ believes } Q\overset{K}{\leftrightarrow}P,\ P \text{ sees } \{X\}_K}{P \text{ believes } Q \text{ said } X},$$

$$\frac{P \text{ believes } \overset{K}{\mapsto}Q,\ P \text{ sees } \{X\}_{K^{-1}}}{P \text{ believes } Q \text{ said } X},$$

$$\frac{P \text{ believes } Q\overset{Y}{\rightleftharpoons}P,\ P \text{ sees } \{X\}_Y}{P \text{ believes } Q \text{ said } X}$$

(2) 新鲜值验证规则

$$\frac{P \text{ believes fresh}(X),\ P \text{ believes } Q \text{ said } X}{P \text{ believes } Q \text{ believes } X}$$

(3) 控制规则

$$\frac{P \text{ believes } Q \text{ controls } X,\ P \text{ believes } Q \text{ believes } X}{P \text{ believes } X}$$

(4) 看到规则

$$\frac{P \text{ sees}(X,Y)}{P \text{ sees } X},\quad \frac{P \text{ sees} < X >_Y}{P \text{ sees } X},\quad \frac{P \text{ believes } Q\overset{K}{\leftrightarrow}P,\ P \text{ sees } \{X\}_K}{P \text{ sees } X},$$

$$\frac{P \text{ believes } \overset{K}{\mapsto}P,\ P \text{ sees } \{X\}_K}{P \text{ sees } X},\quad \frac{P \text{ believes } \overset{K}{\mapsto}Q,\ P \text{ sees } \{X\}_{K^{-1}}}{P \text{ sees } X}$$

(5) 新鲜性规则

$$\frac{P \text{ believes fresh}(X)}{P \text{ believes fresh}(X,Y)}$$

4.1.3 分析实例

利用 BAN 逻辑分析协议的步骤如下。

(1) 给出协议目标。

(2) 由最初的协议得到理想化协议。

(3) 给出初始假设。

（4）根据逻辑规则推理协议目标。

下面以 Kerberos 协议为例给出一个验证实例。

1. Kerberos 协议

Kerberos 协议通过一个认证服务器的帮助，在两个主体之间建立了一个共享密钥。以下给出该协议，其中 A、B 分别表示两个主体，S 表示认证服务器，K_{as} 为 A 和 S 的共享密钥，K_{bs} 为 B 和 S 的共享密钥，T_s 和 T_a 分别为 S 和 A 生成的时间戳，L 为 S 生成的生命周期，第 4 个消息仅在需要相互认证时用到。

Message 1. $A \to S$: A, B.

Message 2. $S \to A$: $\{T_s, L, K_{ab}, B, \{T_s, L, K_{ab}, A\}_{K_{bs}}\}_{K_{as}}$.

Message 3. $A \to B$: $\{T_s, L, K_{ab}, A\}_{K_{bs}}, \{A, T_a\}_{K_{ab}}$.

Message 4. $B \to A$: $\{T_a+1\}_{K_{ab}}$.

2. 协议理想化

对 Kerberos 协议进行理想化可得到：

Message 2. $S \to A$: $\{T_s, A \xleftrightarrow{K_{ab}} B, \{T_s, A \xleftrightarrow{K_{ab}} B\}_{K_{bs}}\}_{K_{as}}$.

Message 3. $A \to B$: $\{T_s, A \xleftrightarrow{K_{ab}} B\}_{K_{bs}}, \{T_a, A \xleftrightarrow{K_{ab}} B\}_{K_{ab}}$ from A.

Message 4. $B \to A$: $\{T_a, A \xleftrightarrow{K_{ab}} B\}_{K_{ab}}$ from B.

3. 协议目标

A believes $A \xleftrightarrow{K_{ab}} B$

B believes $A \xleftrightarrow{K_{ab}} B$

A believes B believes $A \xleftrightarrow{K_{ab}} B$

B believes A believes $A \xleftrightarrow{K_{ab}} B$

4. 初始假设

A believes $A \xleftrightarrow{K_{as}} S$,

B believes $B \xleftrightarrow{K_{bs}} S$,

S believes $A \xleftrightarrow{K_{as}} S$,

S believes $B \xleftrightarrow{K_{bs}} S$,

S believes $A \xleftrightarrow{K_{ab}} B$,

B believes (S controls $A \xleftrightarrow{K} B$),

A believes(S controls $A \xleftrightarrow{K} B$),

B believes fresh(T_s),

A believes fresh(T_s),

B believes fresh(T_a).

在分析 Kerberos 协议时，可将规则应用于以上的假设。这种分析是很直观的，为简单起见，这里只给出关于消息 2 的形式细节，而省略其他细节。

5. 协议分析

A 收到消息 2，则

$$A \text{ sees}\{T_s, (A \xleftrightarrow{K_{ab}} B), \{T_s, A \xleftrightarrow{K_{ab}} B\}_{K_{bs}}\}_{K_{as}}$$

在此之后成立。由于有假设

$$A \text{ believes } A \xleftrightarrow{K_{as}} S$$

应用共享密钥的消息含义规则可得

$$A \text{ believes } S \text{ said } (T_s, A \xleftrightarrow{K_{ab}} B, \{T_s, A \xleftrightarrow{K_{ab}} B\}_{K_{bs}})$$

由组合拆分规则(此处省略)之一可得

$$A \text{ believes } S \text{ said } (T_s, (A \xleftrightarrow{K_{ab}} B))$$

另外,还有以下假设:

$$A \text{ believes fresh}(T_s)$$

根据新鲜值验证规则可得

$$A \text{ believes } S \text{ believes } (T_s, (A \xleftrightarrow{K_{ab}} B))$$

同样,拆分一个组合可得

$$A \text{ believes } S \text{ believes } A \xleftrightarrow{K_{ab}} B$$

然后,将以下假设中的 K 实例化为 K_{ab}

$$A \text{ believes } (S \text{ controls } A \xleftrightarrow{K} B)$$

可以得到更为具体的公式

$$A \text{ believes } (S \text{ controls } A \xleftrightarrow{K_{ab}} B)$$

最后根据控制规则有

$$A \text{ believes } A \xleftrightarrow{K_{ab}} B$$

这就是对消息 2 进行分析所得到的结论。

A 将票据连同另一个包含时间戳的消息传给 B,B 起初只能解密票据:

$$B \text{ believes } A \xleftrightarrow{K_{ab}} B$$

从逻辑上来说,得到这一结果所使用的方法与分析消息 2 时的方法相同,即先使用消息含义规则,再使用新鲜值验证规则,最后使用控制规则。

知道了新的密钥,使得 B 可以解密消息 3 的剩余部分。根据消息含义规则和新鲜值验证规则可得

$$B \text{ believes } A \text{ believes } A \xleftrightarrow{K_{ab}} B$$

第 4 个消息只是简单地向 A 保证了 B 相信密钥,且收到了 A 的最后一个消息,再次将消息含义规则和新鲜值验证规则应用于消息 4,得到最终的结果如下:

$$A \text{ believes } A \xleftrightarrow{K_{ab}} B \qquad B \text{ believes } A \xleftrightarrow{K_{ab}} B$$

$$A \text{ believes } B \text{ believes } A \xleftrightarrow{K_{ab}} B \qquad B \text{ believes } A \text{ believes } A \xleftrightarrow{K_{ab}} B$$

BAN 逻辑主要用来对认证协议进行形式化分析,从上述内容可见,BAN 逻辑非常简洁且分析过程也很简单,这些优点使得 BAN 逻辑在提出以后受到人们的广泛关注。然而,随着进一步的研究,人们发现 BAN 逻辑也还存在不少缺点,如缺乏严格的逻辑语义,协议的理想化过程受人为影响太大等。沿着这一方向,人们又做了进一步研究,形成了 BAN 家族——BAN 类逻辑。如今,人们已经很少直接用 BAN 逻辑本身对协议进行验证了,但其思想对后来的研究甚至未来的研究仍然具有很重要的指导意义。

4.2　Kailar 逻辑

在网络上进行的电子商务与传统商务相比有很多优点，它给商业领域所带来的影响将是革命性的。然而，如果网络上的电子交易不能提供与传统交易相类似的可追究性，那么它无疑会使交易方之间产生争端，从而影响电子商务的应用和发展。Kailar 逻辑[29]便是为分析电子商务协议的可追究性而提出的。

4.2.1　基本术语

Kailar 逻辑可以对协议中主体的证明能力进行分析，即主体证明通信协议中消息来源的能力。协议的主体通常用大写字母 $A, B, \cdots$ 来表示。由每个消息所构造的声明是对该消息的解释，声明通常由小写字母 $x, y, \cdots$ 来表示。在 Kailar 逻辑的分析框架中，声明 x 的证明是为了让其他主体相信该声明而不是让主体自身相信它。根据证明者的证明能力，有以下两种类型的证明。

(1) 强证明：A CanProve x

如果对任意主体 B，主体 A 通过执行一系列操作可以让 B 相信 x，但又不会向 B 泄露任何非 x 的秘密，则称主体 A 能够证明声明 x。

(2) 弱证明：A CanProve x to B

弱证明的含义与强证明类似，只是此处的主体 B 表示一个特定主体而非任意主体。

以下是 Kailar 逻辑中用到的一些表达式：

(1) 签名认证：K Authenticates A

该式表明，密钥 K 可用于认证主体 A 的签名。

(2) 消息解释：x in m

x 是消息 m 的字段或组合字段的解释，这种解释是特定于协议的。

(3) 声明：A Says x

它表示主体 A 对声明 x 以及 x 所蕴涵的声明负责。当主体所说的声明是由两部分组成时，这两部分它都要负责，这一点在下文中将被隐式地使用。

(4) 消息收据：A Receives m SignedWith K^{-1}

它表示主体 A 收到以 K^{-1} 签名的消息。

(5) 信任：A IsTrustedOn x

该式表明，就声明 x 来说，A 是可信的。

4.2.2　逻辑规则

Kailar 逻辑的规则如下：

$$\text{Conj}: \frac{A \text{ CanProve } x;\ A \text{ CanProve } y}{A \text{ CanProve } (x \wedge y)}$$

$$\text{Inf}: \frac{A \text{ CanProve } x;\ x \Rightarrow y}{A \text{ CanProve } y}$$

$$\text{Sign}: \frac{A \text{ Receives } (m \text{ SignedWith } K^{-1});\ x \text{ in } m; A \text{ CanProve } (K \text{ Authenticates } B)}{A \text{ CanProve } (B \text{ Says } x)}$$

$$\text{Trust}: \frac{A \text{ CanProve } (B \text{ Says } x);\ A \text{ CanProve } (B \text{ IsTrustedOn } x)}{A \text{ CanProve } x}$$

4.2.3 分析实例

利用 Kailar 逻辑分析协议的步骤如下。

(1) 给出协议的可追究性目标。

(2) 解释协议消息。

(3) 给出初始假设。

(4) 分析协议的可追究性。

下面以 Medvinsky-Neuman 协议[30]为例给出协议分析实例。通过这个例子可以看出，Kailar 逻辑在协议可追究性分析方面的能力。

1. Medvinsky-Neuman 协议

Message 1. $A \rightarrow B$: K_{AN}

Message 2. $B \rightarrow A$: $\{K_{BN}\}_{K_{AN}}$

Message 3. $A \rightarrow B$: $\{\{\text{coins}\}_{K_{CS}^{-1}},\ SK_{AN1},\ K_{ses},\ S_{id}\}_{K_{BN}}$

Message 4. $B \rightarrow CS$: $\{\{\text{coins}\}_{K_{CS}^{-1}},\ SK_{BN},\ \text{transaction}\}_{K_{CS}}$

Message 5. $CS \rightarrow B$: $\{\{\text{new_coins}\}_{K_{CS}^{-1}}\}_{SK_{BN}}$

Message 6. $B \rightarrow A$: $\{\{\text{amount},\ T_{id},\ \text{date}\}_{K_{BN}^{-1}}\}_{SK_{AN1}}$

其中公钥用 K 加上其拥有者名字的下标来表示，如果下标以 N 结尾，表明该公钥是新生成的，并未到处广播。对称加密系统的密钥用 SK 加个下标来表示。协议中的 transaction 表示主体间的货币交易。在该协议开始时的消息 1 和消息 2 中，付款人 A 得到收款人 B 的密钥，收款人使用该密钥保持其身份的匿名。在消息 3 中，A 发送了款项、希望提供的服务以及两个密钥 SK_{AN1}、K_{ses}，这些均用 B 所提供的密钥进行了加密。

使用 K_{CS}，B 可以验证所收货币是有效的(即它是由有资质的货币服务方提供的)。会话密钥 K_{ses} 主要用来使 B 在服务过程中识别 A。B 在收到货币后通过消息 4 和消息 5 来验证货币的有效性，如果货币还未用，服务方就通过消息 5 向 B 发出新的货币(由于 B 所提供的 A 发来的货币可能是由其他货币服务方发行的，因此，新货币中可将其转换为本地货币)，在消息 6 中，B 返回一个用其私钥签名的收据，该收据同时用 SK_{AN1} 加密以保密。收据包含了支付货币的数目、日期以及一个唯一标识 T_{id}，该标识与会话密钥一起将用于获取服务。

2. 协议目标

该协议有多个目标，此处主要关心其可追究性。

G_1: B CanProve (A Says coins_valid)

G_2: A CanProve (B Says receipt of payment)

3. 协议解释

消息 1、2、3 没签名，所以不用解释。另外，此处的分析只关心付款方与收款方，而消息

4 是由货币服务方接收的，所以也不对其进行解释。

Message 5. B Receives (coins valid) SignedWith K_{CS}^{-1}

Message 6. A Receives (receipt of payment) SignedWith K_{BN}^{-1}

4. 初始假设

A_1: A, B, CS CanProve (K_{CS} Authenticates CS)

A_2: B CanProve (CS IsTrustedOn coins_valid)

5. 协议分析

根据消息 5 的解释，初始假设 A_1，并应用 Sign 规则可得

B CanProve (CS Says coins_valid)

根据初始假设 A_2，应用 Trust 规则可得

B CanProve (coins_valid)

继续下去也无法得到 G_1，从而 G_1 无法得到证明。

再看消息 6。正如 Medvinsky-Neuman 协议的作者所指出的那样，如果 A 收不到消息 6，那么 A 就不能证明 B 收到了款，如果 B 不诚实，他尽可以收了款却不开收据。然而，即使 A 收到了消息 6，下面的分析表明，它还有其他问题存在，即 G_2 不成立，从而 A 无法追究 B 对其提供服务所应负的责任。

针对消息 6 的解释，无法应用 Sign 规则以推出

A CanProve (B Says receipt of payment)

因为要对消息 6 的解释应用 Sign 规则，还需要条件

A CanProve (K_{BN}^{-1} Authenticates B)

而协议中的密钥 K_{BN}^{-1} 是由 B 自由选择的一个密钥，并不能用于认证任何主体。从而 G_2 也不满足。

当然，Medvinsky-Neuman 协议声称要保持协议的匿名性，以上分析表明，该协议不满足可追究性，这一点与匿名性是一致的，但当要求交易满足可追究性时，Medvinsky-Neuman 协议就不能满足要求。

在大多数电子商务交易中，可追究性是一种重要的需求。因为缺乏充分的证据很容易引发交易方之间的争端。Kailar 逻辑为可追究性的分析提供了一种系统的方法，也为协议设计者和分析者对消息进行显式地解释提供了指导。需要指出的是，Kailar 逻辑只是用来分析电子商务协议可追究性的一种逻辑，它并不能解决电子商务协议中的所有安全问题。

4.3 归纳定理证明方法

基于定理证明的形式化验证方法是对协议进行建模或对协议需满足的性质进行形式化，应用定理证明的技术来证明性质是否在协议模型中被满足。协议可以描述成一系列的规则 Γ，协议需满足的性质被描述成另一个公式 Φ，然后证明 $\Gamma \Rightarrow \Phi$。其目的是证明协议是否满足某安全性质，从而辅助寻找对协议的攻击。一般的定理证明方法有 Paulson 归纳法、串空间模型和秩函数等。

4.3.1 归纳定理证明方法概述

简单来说,归纳定理证明方法的基本思想就是通过考察一系列的可能或不可能发生之类的问题来实现证明的目的。下面是一个假设的对话例子。

推销员:协议运行结束后,只有 A 和 B 能够知道会话密钥 K_{ab}。

顾客:偷听者能得到吗?

推销员:因为他不能获得 A 或 B 的长期密钥,所以他不能获得 K_{ab}。

顾客:那么敌手能够欺骗 B 使得他和 B 分享一个密钥吗?

推销员:因为时限值的使用,避免了这种情况的发生。

推销员向顾客推销协议时,声称该协议的会话密钥不会被敌手获得,但是顾客也许会提出他所担心的问题,推销员针对顾客的问题给出相应的回答。当推销员解答了顾客提出的所有问题之后,顾客就可以相信推销员的话了。

归纳定理证明方法中的一个必要的形式化工具就是归纳定义。每个归纳定义都列举出了所有可能的协议的动作,而对应的归纳规则可以让我们推理出任意有限序列的这些动作所造成的结果。通过对结果的逐一考察来完成定理的证明。

利用归纳定理证明方法来验证协议的方法由 Paulson 首先提出,下文中称之为 Paulson 归纳法。Paulson 归纳法假设有一个 Dolev-Yao 模型下的敌手,协议主体的每一个动作定义为一个事件,事件的有序集合称为迹,根据协议运行规则得到的迹的集合称为协议模型。验证协议满足一个性质,即是验证协议模型中的所有迹都满足这个性质。

一般地,基于归纳定理证明的验证方法可以分为以下 3 步。

(1) 用逻辑符号对协议建模。

(2) 用逻辑语言表达出协议需满足的性质。

(3) 使用归纳证明方法来证明性质是否满足。

第(1)步根据协议描述主体可能的行为,它们是通过一条条规则来表达的,这些规则是生成协议模型的基础。第(2)步形式化地描述出协议需要证明的性质。第(3)步使用归纳证明方法对性质进行证明,验证由归纳规则得到的所有迹都满足该性质。证明时需要用到定理证明器 Isabelle。

定理证明方法比其他的协议分析方法,如模型检测,其最大的优点是避免了模型检测中状态爆炸的问题,但同时也带来了其他问题:证明过程难以完全自动化,使用如 Isabelle 这样的定理证明器需要人机交互,并需要人们具有相当强的专业知识,直接给证明带来很大的困难。需要指出的是,即使没有证明出结论来,也不能断然说结论是错误的,很可能是因为证明手段不够熟练导致的。

1. 协议基本元素形式化

主体是协议中最基本的元素,它包括诚实的主体,敌手和可信第三方服务器。nat 表示自然数,对于自然数 i,Friend i 就表示一个特定的主体。Server 表示服务器,Spy 表示协议运行中的敌手。主体形式化为

datatype agent=Server | Friend nat | Spy

密钥在协议中实际上可以看作是一个自然数,密钥形式化为

datatype key=nat

shrK 是一个由主体映射到密钥上的函数，表示主体的长期密钥，如 shrK A 就代表主体 A 的长期密钥。

主体之间传递的任何东西都可以看成消息。它可以做以下递归定义，首先主体、时限值、密钥都是消息，另外消息的组合、消息的 Hash 值和用 key 加密的消息也是新的消息。定义如下：

```
datatype msg= Agent agent
            | Nonce nat
            | Key key
            | Mpair msg msg
            | Hash msg
            | Crypt key msg
```

事件是指主体的一个动作。在 Paulson 归纳法中，允许主体有两个动作，一个是主体向另一个主体发送消息 msg，另一个是主体记录消息 msg。事件形式化为：

```
datatype event =Says agent agent msg
               |Notes agent msg
```

在 Paulson 归纳法中，时间戳被形式化为当前迹的长度。CT：event list→nat 是一个把迹映射到自然数的函数，用来生成当前时间戳，也就是当前迹的长度。CT evs 表示迹 evs 的当前时间戳。

expiredK 用来表示会话密钥已经过期，Tk 是会话密钥生成时的时间戳，sesKlife 是会话密钥的生命周期。如果当前的时间戳减去会话密钥生成时的时间戳 Tk 大于会话密钥的生命周期，说明该会话密钥已经过期，不能接受：

$$\text{expiredK Tk evs}=(\text{CT evs})-\text{Tk}>\text{sesKlife}$$

expiredA 用来表示认证子已经过期，Ta 是认证子生成时的时间戳，authlife 是认证子的生命周期。如果当前的时间戳减去认证子生成时的时间戳 Ta 大于认证子的生命周期，说明该认证子已经过期，不能接受：

$$\text{expiredA Ta evs}=(\text{CT evs})-\text{Ta}>\text{authlife}$$

2. 迹

迹是 Paulson 归纳法中一个非常重要的概念。由事件构成的任意长度的事件有序集合称为迹，它是构成协议模型的基础。一条迹可能描述了协议运行时在信道上发生的所有事件，也就形式化地描述了协议。迹是有序的，并且是反序的，最近发生的事件排在最前边。注意，迹可能是无效的，协议不可能产生的事件列表是不予考虑的，协议模型里仅包括协议有可能产生的迹。

3. 敌手知识集

在 Paulson 归纳法中，定义敌手为 Dolev-Yao 模型下的敌手，首先敌手是一个合法的主体，能够诚实地执行协议。然后敌手控制网络信道，能够窃听、截获和修改信道上的任何消息，也可以向任何主体发送它已知的消息。最后敌手可以对消息做任何操作，比如分开本来连接着的消息，解密它知道密钥的密文消息，或者根据已知消息生成新的消息，但是它不能在不知道密钥的情况下对密文消息解密。

为了表述敌手的能力，Paulson 归纳法引入知识集的概念。敌手知识集是指敌手在当前迹中所拥有的知识。bad 是被腐化主体的集合，它会将长期密钥和私钥泄露给敌手。敌手 Spy 总是被认为是属于集合 bad 的。initState Spy 表示协议运行前敌手的知识集。它包括被腐化主体的长期密钥，被腐化主体的加密私钥和签名私钥，以及所有主体的加密公钥和签名公钥。形式化为：

$$\begin{aligned}\text{initState Spy} = &(\text{Key shrK bad}) \cup \\ &(\text{Key priEk bad}) \cup (\text{Key priSk bad}) \cup \\ &(\text{Key range pubEK}) \cup (\text{Key range pubSK})\end{aligned}$$

仅有初始知识集是不够的，随着协议的运行，敌手的知识会越来越多。使用 spies 函数来描述敌手的动态知识集。spies 是一个把迹映射到消息集合的函数 spies：event list→msg set，用来描述在特定的迹中敌手的知识集。evs 表示迹，spies 的递归定义如下。

对于空迹，敌手的知识集为初始知识集：

$$\text{spies } [\ ] = \text{initState Spy}$$

Says A B X 事件扩展原有的迹 evs 形成新的迹，敌手知道信道上任何主体发送的消息，所以敌手在新迹中的知识集增加了消息 X：

$$\text{spies } ((\text{Says } A\ B\ X) \# \text{evs}) = \{X\} \cup \text{spies evs}$$

Notes A X 事件扩展原有的迹 evs 形成新的迹，敌手知道被腐化主体记录的任何消息，所以如果主体 A 是被腐化的，那么敌手在新迹中的知识集会增加消息 X：

$$\text{spies } ((\text{Notes } A\ X) \# \text{evs}) = \begin{cases} \{X\} \cup \text{spies evs} & \text{若 } A \notin \text{bad} \\ \text{spies evs} & \text{其他} \end{cases}$$

4. analz、synth、parts、used 操作符

仅由以上的知识集来描述敌手的能力还是不够的，因为敌手能够拆分消息、合成消息、解密已知密钥的密文等，故引入 analz、synth、parts、used 操作符。

analz：msg set→ msg set 是一个把消息集合映射到消息集合的函数，用来表示从已知的消息集合里分析出新的消息。H 是一个消息集合，analz 递归定义如下。

（1）消息集合 H 中的任意元素属于 analz H：

$$X \in H \Rightarrow X \in \text{analz } H$$

（2）如果连接的消息$\{X,Y\}$属于 analz H，那么其中的每个消息也属于 analz H：

$$\{X,Y\} \in \text{analz } H \Rightarrow X \in \text{analz } H$$

$$\{X,Y\} \in \text{analz } H \Rightarrow Y \in \text{analz } H$$

（3）如果用密钥 K 加密的消息 X 属于 analz H，并且密钥 K 的解密钥也属于 analz H，那么消息 X 属于 analz H：

$$\text{Crypt } K\ X \in \text{analz } H;\ \text{Key}(\text{invKey } K) \in \text{analz } H \Rightarrow X \in \text{analz } H$$

synth：msg set→msg set 是一个把消息集合映射到消息集合的函数，用来表示从已知的消息集合中合成新的消息。H 是一个消息集合，synth 递归定义如下。

（1）消息集合 H 中的任意元素属于 synth H：

$$X \in H \Rightarrow X \in \text{synth } H$$

（2）任意主体都属于 synth H：

$$\text{Agent } A \in \text{synth } H$$

(3) 如果消息 X 属于 synth H，那么它的 Hash 值也属于 synth H：

$$X \in \text{synth } H \Rightarrow \text{Hash } X \in \text{synth } H$$

(4) 如果消息 X 和 Y 都属于 synth H，那么合成的消息$\{X,Y\}$也属于 synth X：

$$X \in \text{synth } H; \quad Y \in \text{synth } H \Rightarrow \{X,Y\} \in \text{synth } H$$

(5) 如果密钥 K 和消息 X 属于 synth H，那么合成的消息 Crypt K X 也属于 synth H：

$$\text{Key } K \in H; \quad X \in \text{synth } H \Rightarrow \text{Crypt } K\ X \in \text{synth } H$$

parts：msg set→ msg set 是一个把消息集合映射到消息集合的函数，用来表示从已知的消息集合映射和解密出的所有的消息，也就是信道上出现的所有消息的集合。H 是一个消息集合，parts 递归定义如下：

(1) 消息集合 H 的任意元素属于 parts H：

$$X \in H \Rightarrow X \in \text{parts } H$$

(2) 如果连接的消息$\{X,Y\}$属于 parts H，那么其中的每个消息也属于 parts H：

$$\{X,Y\} \in \text{parts } H \Rightarrow X \in \text{parts } H$$

$$\{X,Y\} \in \text{parts } H \Rightarrow Y \in \text{parts } H$$

(3) 如果用密钥 K 加密的消息 X 属于 parts H，那么消息 X 也属于 parts H：

$$\text{Crypt } K\ X \in \text{parts } H \Rightarrow X \in \text{parts } H$$

很明显，parts 操作强于 analz 操作，analz $H \subseteq$ parts H。

used：event list→msg set 是一个把迹映射到消息集合的函数，表示该迹中已经使用过的消息集合，用以支持新鲜性的验证。evs 是一个迹，used 的递归定义如下：

(1) 空迹的已使用消息集合为所有主体的初始知识集的 parts 集合：

$$\text{used}[\,] = \bigcup B.\ \text{parts}(\text{initState}B)$$

(2) 事件 Says A B X 扩展原有的迹 evs 形成新迹，可知消息 X 在新迹中使用过，新迹的 used 集合需要增加进 X 的 parts 集合：

$$\text{used}((\text{Says } A\ B\ X)\ \#\ \text{evs}) = \text{parts}\{X\} \cup \text{used evs}$$

(3) 事件 Notes A X 扩展原有的 evs 形成新迹，可知消息 X 在新迹中使用过，新迹的 used 集合需要增加进 X 的 parts 集合：

$$\text{used}((\text{Notes } A\ X)\ \#\ \text{evs}) = \text{parts}\{X\} \cup \text{used evs}$$

5. 协议模型

形式化协议模型是协议所有可能生成的迹的集合，每一个迹都形式化了协议可能的运行情况。协议模型是归纳定义出的。以自然数 N 的归纳定义为例，$0 \in N$ 并且 $n \in N \Rightarrow \text{Suc } n \in N$，这样就定义出了所有的自然数，协议模型定义方法与之类似。

首先空迹是属于协议模型的。然后利用归纳规则，每个归纳规则形式化协议一步，这些归纳规则说明了怎么用一个事件扩展协议模型里原有的迹形成新的迹，新的迹也是属于协议模型的。如果协议有 n 步，那么它的模型里至少包含 n 个归纳规则，每一条都引入一个 Says 事件。比如协议里仅包括一步：$A \to B$ ：Na，则该步的归纳规则为

$$\llbracket \text{evs} \in \text{dsp}; \text{Nonce Na} \notin \text{used evs} \rrbracket$$

$$\Rightarrow \text{Says } A\ B\ \{\text{Nonce Na}\}\ \#\ \text{evs} \in \text{dsp}$$

dsp 表示协议模型，evs 是原有的属于协议模型的迹，如果时限值 Na 在原来的迹 evs 中没有被使用过，那么 Says 事件可以扩展迹 evs 形成新的迹，新迹是属于协议模型的。

除了上述的归纳规则，Fake 规则允许敌手向任何主体发送它伪造的消息：如果迹 evsF 是属于协议模型的，并且消息 $X \in$ synth（analz（spies evs）），那么事件 Says Spy B X 扩展原迹 evsF 形成的新迹也属于协议模型。可以看出，该规则是符合威胁模型里定义的敌手行为的。形式化如下：

$$⟦\text{evsF} \in \text{dsp}; X \in \text{synth}(\text{analz}(\text{spies evsF}))⟧$$
$$\Rightarrow \text{Says Spy } B\ X\ \#\ \text{evsF} \in \text{dsp}$$

另外，Oops 规则是一个很重要的规则。它形式化了协议运行中可能发生的会话密钥泄露情况，这样敌手就获得了会话密钥。如果迹 evsO 属于协议模型，并且迹 evsO 中发生过会话密钥分配事件，那么事件 Notes Spy {Key Kab} 可以扩展原迹 evsO 形成新的迹，新迹也是属于协议模型的。假设协议包含 $B \rightarrow A$：$\{N_a, K_{ab}\}_{sK_b}$ 的一步，B 向 A 分配会话密钥 K_{ab}，Oops 规则形式化如下：

$$⟦\text{evsO} \in \text{dsp}; \text{Says } B\ A\ (\text{Crypt}(\text{priSK } B)\{\text{Nonce } N_a, \text{Key K}\}) \in \text{set evsO}⟧$$
$$\Rightarrow \text{Notes Spy } \{\text{Nonce } N_a,\ \text{Key } K\}\ \#\ \text{evsO} \in \text{dsp}$$

总之，这些规则归纳定义了如何从一个空迹出发定义出所有可能生成的迹。也就是协议模型的定义。

6. 归纳证明

Paulson 归纳法使用归纳来推证一个归纳定义集合。首先来看数学归纳法，对于每个自然数 n，证明 $P(n)$ 是成立的，P 是某一性质。需证明 $P(0)$ 成立，并且对于每个 $x \in N$，$P(x) \Rightarrow P(\text{Suc } x)$。同样，对于协议模型，需证明 $P([\ ])$ 是成立的，[ ]代表空迹，然后证明如果 $P(\text{evs})$ 是成立的，并且对于所有归纳规则由事件 ev 扩展 evs 形成的新迹 ev # evs，$P(\text{ev0}\ \#\ \text{evs})$ 也是成立的，那么就表明协议模型里所有的迹都满足 P 的性质，也就证明了协议是满足该性质的。

4.3.2 Paulson 归纳法验证 BAN Kerberos 协议的密钥机密性的实例

1. BAN Kerberos 协议模型

BAN Kerberos 协议是一个密钥分配协议。其执行步骤如下。

(1) $A \rightarrow S$：A, B。

(2) $S \rightarrow A$：$\{T_k, B, K_{ab}, \underbrace{\{T_k, A, K_{ab}\}_{K_b}}_{\text{ticket}}\}_{K_a}$。

(3) $A \rightarrow B$：$\underbrace{\{T_k, A, K_{ab}\}_{K_b}}_{\text{ticket}}, \underbrace{\{A, T_a\}_{K_{ab}}}_{\text{authenticator}}$。

(4) $B \rightarrow A$：$\{T_a + 1\}_{K_{ab}}$。

第(1)步，主体 A 首先向服务器 S 请求要和主体 B 建立会话，主体 A 把 A 和 B 的身份标识发给了服务器 S。第(2)步，在服务器 S 收到 A 的请求后，生成一个新鲜的会话密钥 K_{ab} 和当时的时间戳 T_k。$\{T_k, A, K_{ab}\}_{K_b}$ 是票据 Ticket，用主体 B 的长期密钥加密。然后把 $\{T_k, B, K_{ab}, \text{Ticket}\}$ 用 A 的长期密钥加密发送给主体 A。第(3)步，主体 A 收到从 S 发送来的消息后用长期密钥 K_a 解密消息，验证时间戳 T_k 是否过期，主体 B 是否是要想通信的主体。如果无错的话则接受会话密钥 K_{ab}，然后生成当时的时间戳 T_a，把接收来的票据 $\{T_k, A, K_{ab}\}_{K_b}$ 和生成的认证子 $\{A, T_a\}_{K_{ab}}$ 发送给主体 B。第(4)步，主体 B 收到主体 A 发送

来的消息后，解密票据$\{T_k, A, K_{ab}\}_{K_b}$，获得会话密钥K_{ab}。然后解密认证子$\{A, T_a\}_{K_{ab}}$，验证时间戳T_a是否过期。如果正确的话返回给主体A用K_{ab}加密的消息$\{T_a+1\}$。在协议完成后，协议双方A和B都获得了会话密钥K_{ab}。

下面是对 BAN Kerberos 协议形式化建模的规则。

Nil：

[]∈bankerberos

Fake：

⟦evsF∈ bankerberos；X ∈ synth (analz (spies evsF))⟧

⇒Says Spy B X # evsF ∈ bankerberos

BK1：

⟦evs1 ∈ bankerberos⟧

⇒Says A Server ⦃Agent A，Agent B⦄# evsl ∈ bankerberos

BK2：

⟦evs2 ∈ bankerberos；Key K_{ab} ∉ used evs2；K_{ab} ∈ symKeys；

Says A'Server ⦃Agent A，Agent B⦄ ∈ set evs2⟧

⇒Says Server A (Crypt (shrK A) ⦃Number (CT evs2)，Agent B，Key K_{ab}，

Crypt (shrK B) ⦃Number (CT evs2)，Agent A，Key K_{ab}⦄⦄)

evs2∈ bankerberos

BK3：

⟦evs3∈bankerberos；

Says A Server ⦃Agent A，Agent B⦄∈ set evs3；

Says S A(Crypt (shrK A)⦃Number T_s，Agent B，Key K_{ab}，Ticket⦄)

∈ set evs3；

¬expiredK T_s evs3⟧

⇒Says A B ⦃Ticket，Crypt K_{ab}⦃Agent A，Number(CT evs3)⦄⦄

evs3 ∈ bankerberos

BK4：

⟦evs4 ∈ bankerberos；

Says A'B⦃Crypt (shrK B) ⦃Number T_s，Agent B，Key K_{ab}⦄，

Crypt K_{ab}⦃Agent A，Number T_a⦄⦄ ∈ set evs4；

¬expiredK T_s evs4；¬expiredA T_a evs4⟧

⇒Says B A (Crypt K_{ab}(Number T_a))

evs4 ∈ bankerberos

Oops：

⟦evsO ∈ bankerberos；

Says Server A (Crypt (shrK A)⦃Number T_s，Agent B，Key K_{ab}，Ticket⦄)

∈ set evsO⟧

$\Rightarrow$ Notes Spy $\{\!|$Number T_s, Key $K_{ab}|\!\}$

evsO∈bankerberos

其中 bankerberos 是协议模型，是迹的集合。它是由图中的 7 条规则归纳定义的。

Nil 规则指明空迹是属于协议模型的，是归纳定义的基本规则。

Fake 规则允许敌手可以伪造消息并发送给任意主体。如果迹 evsF 属于协议模型，并且消息 X 是敌手能够根据信道上以前的消息伪造出的新的消息，那么敌手可以把它发送给任意主体，Says Spy B X # evsF 形成的新迹也属于协议模型，符号 # 表示事件的连接。

BK1 规则描述了协议的第一步，允许任意主体可以在任何时候开始协议会话。如果迹 evs1 属于协议模型，那么 Says A Server $\{A, B\}$ # bankerberos 形成的新迹也属于协议模型。

BK2 规则描述了协议的第二步，如果迹 evs2 属于协议模型，K_{ab}是没有使用过的对称密钥，这里使用 used 操作实现密钥的新鲜性，并且服务器 Server 收到了请求会话的消息，而不管是哪个主体发送给它的，那么 Server 把第二步中的消息发送给 A，事件扩展形成的新迹也属于协议模型。

BK3 规则描述了协议的第三步，如果迹 evs3 属于协议模型，主体 A 曾向 Server 发送过和 B 建立会话的请求，Server 曾回复过主体 A 第二步的消息，并且时间戳 T_s 没有过期，那么主体 A 向 B 发送第三步的消息，事件扩展形成的新迹也属于协议模型。

BK4 规则描述了协议的第四步，如果 evs4 属于协议模型，B 曾经收到过第三步的消息，不管是哪个主体发给它的，并且时间戳 T_s、T_a 都没有过期，那么 B 向 A 回复第四步的消息，事件扩展形成的新迹也属于协议模型。

Oops 规则形式化了在协议运行中可能因为某些原因导致会话密钥的泄露，如果迹 evs0 属于协议模型，并且 Server 曾向 A 发送过第二步的消息，那么敌手有可能记录下会话密钥 K_{ab}，事件扩展形成的新的迹也属于协议模型。

2. 密钥机密性

密钥机密性是指在协议运行完毕后除了协议双方和可信第三方没有另一个主体能够获得会话密钥。BAN Kerberos 协议的机密性可描述为，如果主体 A 和 B 都没有被腐化，服务器 Server 曾向主体 A 分配过会话密钥 K_{ab}，并且 K_{ab}没有因为 Oops 规则而泄露，那么敌手是无法获得会话密钥 K_{ab}的。形式化为：

如果 A 和 B 都没有被腐化，并且

Says Server A(Crypt (shrK A\{NumberT_k, Agent B, Key K_{ab}, Ticket\}))∈evs,

Notes Spy\{Number T_k, Key K_{ab}\}∉evs,

那么 Key K_{ab}∉analz(spies evs)。

直观上，如果 A、B 都没有被腐化，则敌手无法获得 A、B 的长期密钥，也就无法解密包含会话密钥 K_{ab}的密文，又 K_{ab}没有因为 Oops 规则而泄露，所以敌手是无法得知 K_{ab}的。具体证明该定理时，需要用到定理证明器 Isabelle 进行归纳证明，首先空迹是满足该性质的，然后运用归纳规则生成新的迹，证明新迹也是满足该性质的。于是协议模型中的所有迹都

是满足该性质的，也就是说，协议模型是满足该性质的，定理得证。

4.4 应用 Pi 演算方法

安全协议验证已经成为许多研究人员所关注的问题。该问题之所以如此重要，是因为协议的设计太容易出错了，而且，这些错误仅在恶意敌手存在的情况下出现，很难用测试的方法检测到。在该领域中，一种重要的趋势是：基于 Dolev-Yao 模型，在无穷会话的条件下，且尽可能少地依赖于人为参与来验证协议。由于协议的不安全性对于有限会话来说是一个 NP 完全问题，对于无穷会话来说，是不可判定的。从而，并不是对所有协议进行无穷会话下的自动验证都可完成。利用基于语言的技术，如分类或抽象解释等，通过可靠的近似来处理无限状态系统，可完成一些典型的验证。这些技术是不完备的（正确的协议可能无法通过类型检查，或者抽象解释工具所发现的可能是错误的攻击），但却是可靠的（如果它没有发现攻击，则可以保证协议满足相应的属性）。这一点对协议的评判来说是非常重要的。

本节的目标是通过提供一种完全自动的技术来验证具有无穷会话的安全协议的对应性，从而扩展前人在该领域的工作。对应性是具有以下形式的一种属性：如果协议执行了一些事件，那么它必须在此之前已经执行过另外一些事件。这里使用了可描述对应性的一种丰富的语言，其中，通过含有合取和析取的逻辑公式，便可描述那些要求必须已执行过的事件。另外，还考虑了非单射对应性（如果协议执行了一些事件，那么它必须在此之前已经执行过另外一些事件至少一次）和单射对应性（如果协议执行了一些事件 n 次，那么它必须在此之前已经执行过另外一些事件至少 n 次）。对应性一开始被命名为对应性断言，类似的名称还有最初用于建模认证性的协定（Agreement）。直观地说，如果 B 希望交谈的对象是 A，那么要保证他实际交谈的对象就是 A，就需要协议对 B 来认证 A。当 B 认为他已经和 A 运行过协议时就执行事件 $e(A, B)$，当 A 认为她和 B 运行协议时，她执行另一事件 $e'(A,B)$，如果 B 执行事件 $e(A, B)$时，A 已经执行过 $e'(A,B)$，那么认证性就得到了满足。文献中也有些类似这一方案的变体，下文将表明，该技术可处理大部分这类变体。也可按以下方法用对应性描述保密性。称值 M 在协议中是保密的，是指敌手不能获得值 M，如果将事件 attacker(M)与敌手获取 M 这一事实关联起来，那么 M 的保密性就可描述为“不能执行 attacker(M)”，也就是说，“如果执行了 attacker(M)，那么保密性为假”。更复杂的属性也可用对应性的概念来描述，如“协议的所有消息已被按序发送”等。

该技术基于前人关于保密性验证技术的重要扩展。具体来说，协议用进程演算来描述，它在 pi 演算中加入了密码原语。该进程演算将事件扩充进来，用以描述对应性。这些事件只是用来表示协议，而不用于证明对应性。然后协议被自动转换为一组 Horn 子句。最后，将这些子句传给一个基于归结（Resolution-based）的求解器。该求解器并非总是终止的，但它对一种经过标记的协议来说是终止的。实验结果表明，它对很多协议都是终止的。

该方法的优点是：它是完全自动的；用户只需对协议和要证的对应性进行编码即可。它

不限制协议会话的数目及敌手可处理的项的规模。它可处理很通用的密码原语，包括共享密钥加密、公钥加密、签名、单向 Hash 函数及 Diffie-Hellman 密钥协商。它有明确的语义基础。缺点是个别时候算法不终止，同时该技术是不完备的：向 Horn 子句的转换引入了一种抽象，该抽象无法记住每个行为副本的编号。这种抽象对于处理无穷会话来说是关键的，也正是这种抽象提供了证明对应性的充分条件，但对正确协议可能会失败。基本上，在证明那些开始时需要对一些值保密，之后又需要公开这些值的协议时会失败。在实践中，该工具仍然是很精确的。在实验中，在证明那些正确的协议时它总是成功的。

4.4.1 进程演算

本节给出用于描述安全协议的进程演算，包括其语法、语义及一个协议描述实例。

1. 语法及非形式化语义

表 4.1 给出了本演算中项（数据）和进程（程序）的语法，用 a、b、c、k 及类似的标识符表示名字（Names），用 x、y、z 表示变量。该语法还采用了一组符号表示构造子和析构子。通常用 f 表示构造子，用 g 表示析构子。

表 4.1　进程代数语法

$M,N::=$	项
x,y,z	变量
a,b,c,k	名字
$f(M_1,\cdots,M_n)$	构造子应用
$P,Q::=$	进程
$\overline{M}\langle N\rangle.P$	输出
$M(x).P$	输入
0	空
$P\mid Q$	并行组合
$!P$	重复
$(\nu a)P$	限制
let $x=g(M_1,\cdots,M_n)$ in P else Q	析构子应用
if $M=N$ then P else Q	条件
event$(M).P$	事件

构造子用以构造项。因此，项包括变量、名字以及 $f(M_1,\cdots,M_n)$形式的构造子应用；项是无类型的。然而，析构子不会出现在项中，只用来在进程中处理项，是可用于进程中的一种项上的偏函数。在进程 let $x=g(M_1,\cdots,M_n)$in P else Q 中，首先要对 $g(M_1,\cdots,M_n)$求值，如果成功，则将 x 绑定于所得的结果并执行 P，否则执行 Q。更确切地说，一个 n 元析构子 g 的语义由一个重写规则的集合 def(g)给出，规则的形式为 $g(M_1,\cdots,M_n)\rightarrow M$，其中 $M_1,\cdots,M_n$、M 为非名字的项，M 中的变量同样出现在 $M_1,\cdots,M_n$ 中。当且仅当存在一个代换 σ 和 def(g)中的一个重写规则 $g(M_1,\cdots,M_n)\rightarrow M$，使得 $M_i'=\sigma M_i\,(i\in\{1,\cdots,n\})$且 $M'=\sigma M$ 时，可用 $g(M_1',\cdots,M_n')\rightarrow M'$来扩展这些规则。使用构造子和析构子可表示如表 4.2 所示的数据结构和密码操作。

表 4.2　构造子与析构子

	构　造　子	析　构　子
元组	元组 ntuple(x_1, …, x_n)	投影 ith_n(ntuple (x_1, …, x_n)) → x_i
共享密钥加密	用密钥 y 对 x 加密，sencrypt (x, y)	解密 sdecrypt (sencrypt(x, y), y) → x
随机共享密码加密	用密钥 y 及随机因子 r 对 x 加密，sencrypt_p(x, y, r)	解密 sdecrypt_p (sencrypt_p (x, y, r), y) → x
随机公钥加密	用密钥 y 及随机因子 r 对 x 加密，pencrypt_p(x, y, r) 由私钥 y 生成公钥，pk(y)	解密 pdecrypt_p (pencrypt_p (x, y, r), y) → x
签名	用私钥 y 对 x 签名，sign (x, y) 由私钥 y 生成公钥，pk(y)	签名验证 checksignature (sign(x, y), pk (y)) → x 无签名消息 getmessage (sign (x, y)) → x
无消息泄露签名	用私钥 y 对 x 签名，nmrsign (x, y) 由私钥 y 生成公钥，pk(y) 常量 true	验证 nmrchecksign (nmrsign (x, y), pk (y), x) → true
单向 Hash 函数	Hash 函数 $h(x)$	
拥有者名及密码列表	由密钥到其拥有者名 host (x)	由拥有者名到密钥 getkey (host(x)) → x

构造子和析构子可以是公有的，也可以是私有的。公有的可以被敌手使用，如果没有进行其他声明，一般都是指公有的。私有的只能被诚实的参与者使用，在实践中，它们可用于建模，如保存在服务器中的密钥表。公有的构造子 host 由一个长期私钥计算其拥有者的名字，私有的析构子 getkey 从主体的名字返回其密码，模拟在(host name, key)对形式的表中所进行的查找。用公有的构造子 host 允许敌手建立并注册任意数目的主体名及其密码。然而，由于 getkey 是私有的，敌手将不能从一个主体名计算出其密钥，否则将违反所有协议的约定：主体名是公开的，而诚实参与者的密钥则是保密的。

进程演算额外提供了一种执行事件的结构，可用于描述对应性。进程event(M). P 先执行事件 event(M)，然后执行 P。

表 4.1 中的其他结构是标准的，大多数来源于 pi 演算，其中输入进程 $M(x).P$ 在通道 M 上输入一个消息，然后执行 P，其中 P 中的 x 与输入消息绑定。输出进程 $\overline{M}\langle N\rangle.P$ 在通道 M 上输出消息 N，然后执行 P。空进程 0 什么也不做。进程 $P|Q$ 为 P 和 Q 的并发组合。复制!P 表示任意多个 P 的副本在并发运行。限制(va)P 生成一个新的名字，然后执行 P。条件 if $M=N$ then P else Q，当 M 和 N 在运行时可归约为同一个项时执行 P，否则执行 Q。人们将 let $x=M$ in P 定义为 $P\{M/x\}$ 的另一种表达。通常如果 else 子句包括 0 时将其省略。

名字 a 在进程(va)P 中是约束的，在进程 $M(x).P$ 和 let $x=g(M_1,\cdots,M_n)$ in P else Q 中，变量 x 约束于 P。记 $f_n(P)$ 和 $f_v(P)$ 分别为在 P 中自由的名字和变量。如果一个进程没有自由变量，则称其是闭的；它可能有自由名字。

2. 操作语义

语义格局是一个 E、P 对，其中环境 E 是一个名字的有限集，P 为一个闭进程的有限多

重集合。环境 E 必须包括 P 中进程的所有自由名字。直观地看,格局$\{a_1,\cdots,a_n\}$,$\{P_1,\cdots,P_n\}$对应于进程$(\nu a_1)\cdots(\nu a_n)(P_1|\cdots|P_n)$。演算的语义由表 4.3 所示的定义在语义格局上的归约关系→定义。

表 4.3 结构化语义

$E,\mathcal{P}\cup\{0\}\rightarrow E,\mathcal{P}$	(Red Nil)
$E,\mathcal{P}\cup\{!P\}\rightarrow E,\mathcal{P}\cup\{P,\ !P\}$	(Red Repl)
$E,\mathcal{P}\cup\{P\mid Q\}\rightarrow E,\mathcal{P}\cup\{P,\ Q\}$	(Red Par)
$E,\mathcal{P}\cup\{(\nu a)P\}\rightarrow E\cup\{a'\},\mathcal{P}\cup\{P\{a'/a\}\}$	(Red Res)
where $a'\notin E$.	
$E,\mathcal{P}\cup\{\overline{N}\langle M\rangle.\ Q,\ N(x).\ P\}\rightarrow E,\ \mathcal{P}\cup\{Q,\ P\{M/x\}\}$	(Red I/O)
$E,\mathcal{P}\cup\{\text{let } x=g(M_1,\cdots,M_n)\text{in } P \text{ else } Q\}\rightarrow E,\mathcal{P}\cup\{P\{M'/x\}\}$	
if $g(M_1,\cdots,M_n)\rightarrow M'$	(Red Destr 1)
$E,\mathcal{P}\cup\{\text{let } x=g(M_1,\cdots,M_n)\text{ in } P \text{ else } Q\}\rightarrow E,\mathcal{P}\cup\{Q\}$	(Red Destr 2)
If there exists no M' such that $g(M_1,\cdots,M_n)\rightarrow M'$	
$E,\mathcal{P}\cup\{\text{if } M=M \text{ then } P \text{ else } Q\}\rightarrow E,\mathcal{P}\cup\{P\}$	(Red Cond 1)
$E,\mathcal{P}\cup\{\text{if } M=N \text{ then } P \text{ else } Q\}\rightarrow E,\mathcal{P}\cup\{Q\}$	(Red Cond 2)
if $M\neq N$	
$E,\mathcal{P}\cup\{\text{event}(M).\ P\}\rightarrow E,\mathcal{P}\cup\{P\}$	(Red Event)

3. 例子

作为一个运行的例子,考虑 Needham-Schroeder 公钥密码协议的一种简化版本。协议包括以下消息:

$$\text{Message 1.}\ A\rightarrow B:\ \{a,pk_A\}_{pk_B}$$

$$\text{Message 2.}\ B\rightarrow A:\ \{a,b,pk_B\}_{pk_A}$$

$$\text{Message 3.}\ A\rightarrow B:\ \{b\}_{pk_B}$$

A 首先给 B 发送一个用 B 的公钥加密的 nonce(新鲜名)a,B 用其私钥 sk_B 解密该消息并回传 nonce a,他选择的新的 nonce b,以及他自己的公钥 pk_B,这些均用 pk_A 加密。当 A 收到这一消息时对其进行解密。当 A 看到 nonce a,A 相信是 B 的回答,因为只有 B 可以解密第一个关于 a 的消息并获得 a。然后 A 发回用 pk_B 加密的nonce b,B 解密该消息。当 B 看到 nonce b,他相信是 A 的回答,因为只有 A 可解密第二个消息,并获得其中的 b。第一个消息中的 pk_A 和第二个消息中的 pk_B 使得这些消息明显是用于 A 和 B 之间的会话,从而避免中间人攻击。该协议在演算中可由进程 P 来描述,解释如下。

$P_A(sk_A,pk_A,pk_B)=!c(x_pk_B).(\nu a)\text{event}(e_1(pk_A,x_pk_B,a)).$

$(\nu r_1)\bar{c}\langle\text{pencrypt}_p((a,pk_A),x_pk_B,r_1)\rangle.$

$c(m).\text{let}(=a,\ x_b,\ =x_pk_B)\ =\text{pdecrypt}_p(m,\ sk_A)\text{in}$

$\text{event}(e_3(pk_A,x_pk_B,\ a,\ x_b)).(\nu r_3)\bar{c}\langle\text{pencrypt}_p(x_b,x_pk_B,r_3)\rangle$

$\text{if } x_pk_B=pk_B \text{ then}$

$\text{event}(e_A(pk_A,x_pk_B,a,\ x_b)).\bar{c}\langle\text{sencrypt}(sAa,a)\rangle.\bar{c}\langle\text{sencrypt}(sAb,x_b)\rangle$

$P_B(sk_B, pk_B, pk_A) = !c(m').\mathrm{let}(x_a, x_pk_A) = \mathrm{pdecrypt_p}(m', sk_B)\ in\ (vb)$

$\mathrm{event}(e_2(x_pk_A, pk_B, x_a, b)).\ (vr_2)\bar{c}\langle \mathrm{pencrypt_p}((x_a, b, pk_B), x_pk_A, r_2)\rangle.$

$c(m'').\mathrm{let}(=b) = \mathrm{pdecrypt_p}(m'', sk_B)\mathrm{in}$

$\mathrm{if}\ x_pk_A = pk_A\ \mathrm{then}$

$\mathrm{enent}\ (e_B(x_pk_A, pk_B, x_a, b)).\bar{c}\langle \mathrm{sencrypt}(sBa, x_a)\rangle.\bar{c}\langle \mathrm{sencrypt}\ (sBb, b)\rangle$

$P = (vsk_A)(vsk_B)\mathrm{let}pk_A = pk(sk_A)\mathrm{in\ let}pk_B = pk(sk_B)\mathrm{in}$

$\bar{c}\langle pk_A\rangle\bar{c}\langle pk_B\rangle.(P_A(sk_A, pk_A, pk_B) | P_B(sk_B, pk_B, pk_A))$

进程 P 首先创建 A 和 B 的公私钥，并将公钥在通道 c 上输出以建模敌手的初始知识中包含它们，然后协议本身开始执行：P_A 描述 A，P_B 描述 B，两个主体均可运行无限次会话，因此 P_A 和 P_B 以重复开始。

我们认为 A 和 B 均希望与任意主体对话。因此，为了决定 A 将与谁对话，A 首先输入一个包含其对话者公钥 x_pk_B 的消息（该对话者因此可由敌手选定）。然后 A 通过选择 nonce a，并执行事件 $e_1(pk_A, x_pk_B, a)$开始一个协议的运行。直觉上，该事件记录了 A 在有公钥 x_pk_B 参与并使用了 nonce a 的运行中发送协议的消息 1。事件 e_1 被放在实际输出消息 1 之前，这对所期望成立的对应性来说是必要的：如果事件 e_1 被放在实际输出消息 1 之后，那么甚至在已经输出消息 1 的情况下也不能证明 e_1 一定是已被执行了的，因为消息 1 可以在不执行 e_1 的情况下输出。下面的事件 e_2 和 e_3 与此情形类似。然后 A 发送协议的第一个消息 $\mathrm{pencrypt_p}((a, pk_A), x_pk_B, r_1)$，其中 r_1 为新鲜因子，用以建模公钥加密是随机的。A 等待第二个消息并用其私钥 sk_A 解密。如果解密成功，A 使用模式匹配的组成 $\mathrm{let}(=a, x_b, =x_pk_B) = \mathrm{pdecrypt_p}(m, sk_A)\mathrm{in}\cdots$来检查消息的形式是正确的。该组成是以下语法的简写：

$$\mathrm{let}\ y = \mathrm{pdecrypt_D}(m, sk_A)\mathrm{in\ let}\ x_1 = 1th_3(y)\mathrm{in\ let}\ x_b = 2th_3(y)\mathrm{in\ let}\ x_3 = 3th_3(y)\ \mathrm{in\ if}\ x_1 = a\ \mathrm{then\ if}\ x_3 = x_pk_B\ \mathrm{then}\cdots,$$

然后 A 执行事件 $e_3(pk_A, x_pk_B, a, x_b)$，以记录她在有公钥 x_pk_B，nonce a 及 x_b 参与的会话中已收到消息 2 并发送了消息 3。最后她发送协议的最后一个消息 $\mathrm{pencrypt_p}(x_b, x_pk_B, r_3)$。发送完该消息后，$A$ 执行一些仅用于描述协议属性的行为。当 $x_pk_B = pk_B$，也就是说，会话是 A 和 B 之间的会话，A 执行事件 $e_A(pk_A, x_pk_B, a, x_b)$，以记录结束了有公钥 x_pk_B，nonce a 和 x_b 参与的会话。A 还要输出一个用 nonce a 加密的秘密名 sAa 以及用 nonce b 加密的秘密名 sAb，这些输出有助于形式化 nonce 的保密性。用工具可证明自由名的保密性，但不能证明约束名（如 a）或变量（如 x_b）的保密性，为克服这一不足，使用 a 对一个自由名 sAa 进行加密，这样，sAa 是保密的当且仅当由 A 选择的 nonce a 是保密的。类似地，sAb 是保密的当且仅当由 B 选择的 nonce b 是保密的。

类似地，可以对进程 B 的执行进行解释。

事件将用于形式化认证性。例如，若想形式化"如果 A 结束了一个协议的会话，那么 B 已经开始了一个具有相同 nonce 的会话"，便可如期望的那样形式化为：如果 $e_A(x_1, x_2, x_3, x_4)$已被执行，那么 $e_2(x_1, x_2, x_3, x_4)$已被执行。

4.4.2 对应性的定义

本节将形式化地定义所要验证的对应性。

假设协议的运行中存在敌手，且根据 Dolev-Yao 模型，敌手可以监听所有消息，计算并发送它所拥有的所有消息。因此，一个敌手可以用这样一种进程来描述，该进程有一组公有名 Init 在敌手的初始知识中，且不包含事件(虽然初始知识仅包含 Init 中的名字，但通过在 Init 中的通道上的发送行为，可将任何项提供给敌手)。

定义 4.1 令 Init 为名字的有限集。闭进程 Q 为 Init 敌手当且仅当 $f_n(Q) \subseteq \text{Init}$，且 Q 不包含事件。

1. 非单射对应性

接下来，定义什么时候一个迹满足一个原子 α。其中，原子由如表 4.4 所示的语法生成。

表 4.4 原子的语法生成

$\alpha ::=$	原子
attacker(M)	攻击者的知识
message(M,M')	通道上的消息
event(M)	事件

直觉上，一个迹满足 attacker(M)，说明攻击者拥有 M，或者等价地，M 被通过 Init 中的公有通道发送。迹满足 message(M,M')，说明消息 M' 被通过通道 M 发送。迹满足 event(M)，说明事件 event(M)已被执行。

定义 4.2 称一个迹 $\mathcal{T} = E_0, \mathcal{P}_0 \rightarrow^* E', \mathcal{P}'$ 满足 attacker(M)当且仅当存在 $E, \mathcal{P}, x, P, Q, c \in \text{Init}$，$\mathcal{T}$ 包含转换 $E, \mathcal{P} \cup \{\bar{c}\langle M\rangle.Q, c(x).P\} \rightarrow E, \mathcal{P} \cup \{Q, P\{M/x\}\}$。

称一个迹 $\mathcal{T} = E_0, \mathcal{P}_0 \rightarrow^* E', \mathcal{P}'$ 满足 message(M,M')当且仅当存在 $E, \mathcal{P}, x, P, Q$，使得 $\mathcal{T}$ 包含转换 $E, \mathcal{P} \cup \{\overline{M}\langle M'\rangle.Q, M(x).P\} \rightarrow E, \mathcal{P} \cup \{Q, P\{M'/x\}\}$。

称一个迹 $\mathcal{T} = E_0, \mathcal{P}_0 \rightarrow^* E', \mathcal{P}'$ 满足 event(M)当且仅当存在 E、$\mathcal{P}$、P，使得 $\mathcal{T}$ 包含转换 $E, \mathcal{P} \cup \{\text{event}(M).P\} \rightarrow E, \mathcal{P} \cup \{P\}$。

从直觉上看，对应性 $\alpha \Rightarrow \bigvee_{j=1}^{m}(\alpha_j \rightsquigarrow \bigwedge_{k=1}^{l_j} \text{event}(M_{jk}))$ 表示，如果 α 的一个实例被满足，那么存在 $j \in \{1,\cdots,m\}$，使得该实例是 α_j 的一个实例，且每个事件 event(M_{j1})，…，event(M_{jl_j})的相应实例已被执行。

定义 4.3 闭进程 P_0 对于 Init 敌手来说满足对应性

$$\alpha \Rightarrow \bigvee_{j=1}^{m} \left(\alpha_j \rightsquigarrow \bigwedge_{k=1}^{l_j} \text{event}(M_{jk})\right)$$

当且仅当对任意 Init 敌手 Q，对任意包含 $f_n(P_0) \cup \text{Init} \cup f_n(\alpha) \cup \bigcup_j f_n(\alpha_j) \cup \bigcup_{j,k} f_n(M_{jk})$ 的 E_0，对任意代换 σ，对任意迹 $\mathcal{T} = E_0, \{P_0, Q\} \rightarrow^* E', \mathcal{P}'$，如果 $\mathcal{T}$ 满足 $\sigma\alpha$，那么存在 σ'，$j \in \{1,\cdots,m\}$，使得 $\sigma'\alpha_j = \sigma\alpha$，且对所有 $k \in \{1,\cdots,l_j\}$，$\mathcal{T}$ 也满足 event($\sigma' M_{jk}$)。

定义 4.3 非常一般化。下面给出一些有趣的特例。当 $m=0$ 时，析取式 $\bigvee_{j=1}^{m}\cdots$ 被表示为 false，当对所有的 j，$\alpha=\alpha_j$ 时，对应性被缩写为 $\alpha \rightsquigarrow \bigvee_{j=1}^{m} \bigwedge_{k=1}^{l_j} \text{event}(M_{jk})$。该对应性意味着，如果 α 的一个实例被满足，那么存在 $j \leqslant m$，使得事件 event(M_{j1})，…，event(M_{jl_j})的相应实例已被执行。α 中的变量是被全称量化的(因为 σ 是全称量化的)。出现在 M_{jk} 中，但未出现在 α 中的变量是存在量化的(因为 σ' 是存在量化的)。

例 4.1 在 4.4.1 节的进程中，对应性 enent($e_B(x_1,x_2,x_3,x_4)$) ⇝ event($e_1(x_1,x_2,x_3)$) ∧ event($e_2(x_1,x_2,x_3,x_4)$) ∧ event($e_3(x_1,x_2,x_3,x_4)$)的意思是，如果事件 event($e_B(x_1,x_2,x_3,x_4)$)被执行，那么，具有相同参数值 x_1,x_2,x_3,x_4 的事件event($e_1(x_1,x_2,x_3)$)，event($e_2(x_1,x_2,x_3,x_4)$)，event($e_3(x_1,x_2,x_3,x_4)$)均已被执行。

对应性

$$\begin{aligned}&\text{event}(R_\text{received}(msg(x,z)))\Rightarrow\\&\quad(\text{event}(R_\text{received}(msg(x,(z',\text{Auth}))))\rightsquigarrow\\&\quad\text{event}(S_\text{has}(k,msg(x,(z',\text{Auth}))))\wedge\\&\quad\text{event}(\text{TTP_send}(\text{sign}((\text{sencrypt}(msg(x,(z',\text{Auth})),k),x),sk_{\text{TTP}}))))\\&\vee(\text{event}(R_\text{received}(msg(x,(z',\text{NoAuth}))))\rightsquigarrow\\&\quad\text{event}(S_\text{has}(k,msg(x,(z',\text{NoAuth}))))\wedge\\&\quad\text{event}(\text{TTP_send}(\text{sign}(\text{sencrypt}(msg(x,(z',\text{NoAuth})),k),sk_{\text{TTP}}))))\end{aligned}$$

的意思是说，如果事件 R_received($msg(x,z)$)已被执行，那么有两种可能发生：存在 z' 或者 $z=(z',\text{Auth})$ 或者 $z=(z',\text{NoAuth})$，无论哪种情况，都存在 k，使得事件 TTP_send (certificate)和 S_has($k,msg,(x,z)$)已执行，但使用不同的 certificate 值。当 $z=(z',\text{Auth})$时，certificate＝sign((S2TTP, x), | sk_{TTP})，当 $z=(z',\text{NoAuth})$时，certificate＝sign (S2TTP, sk_{TTP})，其中，S2TTP＝sencrypt($msg(x,z),k$)。

以下定义为定义 4.3 的特例。

定义 4.4 闭进程 P 对于 Init 保持 M 的所有实例的秘密性，当且仅当它对于 Init 敌手满足对应性 attacker(M) ⇝ false。

定义 4.5 非单射协定(Agreement)是一种 event($e(x_1,\cdots,x_n)$) ⇝ event($e'(x_1,\cdots,x_n)$)形式的对应性。

2. 单射对应性

定义 4.6 称事件 event(M)在$\mathcal{T}=E_0,\mathcal{P}_0\rightarrow^* E',\mathcal{P}'$的第 τ 步被执行，当且仅当存在 E，$\mathcal{P}$，P，使得$\mathcal{T}$的第 τ 个变换是以下形式的变换：$E,\mathcal{P}\cup\{\text{event}(M).P\}\rightarrow E,\mathcal{P}\cup\{P\}$。

直观地说，一个单射对应性 event(M) ⇝ inj enent(M')要求每个事件 event(σM)被不同的事件 event($\sigma M'$)激发，而非单射对应性 event(M) ⇝ event(M')允许多个事件 event(σM)被同一个事件 event($\sigma M'$)激发。用[inj]表示一个可选标记，它可以取 inj 或什么都不取，当[inj]＝inj 时，要求单射对应性，当[inj]什么都不取时，对应性不需要是单射的。

定义 4.7 闭进程 P_0 对于 Init 敌手来说满足对应性

$$\text{event}(M)\Rightarrow\bigvee_{j=1}^{m}\left(\text{event}(N_j)\rightsquigarrow\bigwedge_{k=1}^{l_j}[\text{inj}]_{jk}\,\text{event}(M_{jk})\right)$$

当且仅当，对任意 Init 敌手 Q，对任意包含 $f_n(P_0)\cup\text{Init}\cup f_n(M)\cup\bigcup_j f_n(N_j)\cup\bigcup_{j,k} f_n(M_{jk})$的 E_0，对任意代换 σ，对任意迹$\mathcal{T}=E_0,\{P_0,Q\}\rightarrow^* E',\mathcal{P}'$，存在从$\mathcal{T}$的步骤集的子集到$\mathcal{T}$的步骤集的函数 ϕ_{jk}，使得：

(1) 对所有的 τ，如果存在 σ，使得 event(σM)在$\mathcal{T}$的第 τ 步被执行，那么，存在 σ' 和 j，使得 $\sigma' N_j=\sigma M$，且对所有的 $k\in\{1,\cdots,l_j\}$，$\phi_{jk}(\tau)$是有定义的，且 event($\sigma' M_{jk}$)在 τ 的第 $\phi_{jk}(\tau)$ 步被执行。

(2) 如果$[\mathrm{inj}]_{jk}=\mathrm{inj}$，则$\phi_{jk}$是单射的。

与非单射对应性类似，当对所有的j有$M=N_j$时，对应性可缩写为

$$\mathrm{event}(M)\rightsquigarrow\bigvee_{j=1}^{m}\bigwedge_{k=1}^{l_j}[\mathrm{inj}]_{jk}\mathrm{event}(M_{jk})$$

注意：当$\alpha=\mathrm{attacker}(M)$时，对应性$\alpha\Rightarrow\bigvee_{j=1}^{m}\left(\alpha_j\rightsquigarrow\bigwedge_{k=1}^{l_j}[\mathrm{inj}]_{jk}\mathrm{event}(M_{jk})\right)$中至少有一个 inj 标记时总会导致错误：敌手可以任意次地在它的一个通道上一直重复输出M。当$\alpha=\mathrm{message}(M,M')$，且至少有一个 inj 标记时，对应性只有在敌手不能执行相应输出时也许为真。为简单起见，我们只关注$\alpha=\mathrm{event}(M)$时的情况。

定义 4.8 单射协定是以下形式的对应性：

$$\mathrm{event}(e(x_1,\cdots,x_n))\rightsquigarrow\mathrm{inj}\ \mathrm{event}(e'(x_1,\cdots,x_n))$$

单射协定要求执行 $\mathrm{event}(e(M_1,\cdots,M_n))$的次数要小于执行 $\mathrm{event}(e'(M_1,\cdots,M_n))$的次数：每次 $\mathrm{event}(e(M_1,\cdots,M_n))$的执行对应于一个不同的 $\mathrm{event}(e'(M_1,\cdots,M_n))$的执行。

3. 一般对应性

对应性还可提供关于事件执行顺序的信息。

首先来看一个例子。使用 4.4.1 节的进程用

$$\begin{aligned}&\mathrm{event}(e_B(x_1,x_2,x_3,x_4))\rightsquigarrow(\mathrm{inj}\ \mathrm{event}(e_3(x_1,x_2,x_3,x_4))\rightsquigarrow\\&(\mathrm{inj}\ \mathrm{event}(e_2(x_1,x_2,x_3,x_4))\rightsquigarrow(\mathrm{inj}\ \mathrm{event}(e_1(x_1,x_2,x_3)))))\end{aligned}\qquad(4\text{-}1)$$

来表示这样一种对应性：每次事件$e_B(x_1,x_2,x_3,x_4)$的执行对应于事件 events $e_1(x_1,x_2,x_3)$，$e_2(x_1,x_2,x_3,x_4)$，$e_3(x_1,x_2,x_3,x_4)$的不同执行，且$e_B(x_1,x_2,x_3,x_4)$的每次执行之前均有不同的事件$e_3(x_1,x_2,x_3,x_4)$执行，每次$e_3(x_1,x_2,x_3,x_4)$本身的执行之前，均有不同的事件$e_2(x_1,x_2,x_3,x_4)$执行，每次$e_2(x_1,x_2,x_3,x_4)$本身的执行之前，均有不同的事件$e_1(x_1,x_2,x_3)$执行。该对应性表明，B终止和A的对话前，A与B已经以期望的顺序交换了协议所有的消息。该对应性并不等价于以下对应性的合取：

$$\mathrm{event}(e_B(x_1,x_2,x_3,x_4))\rightsquigarrow\mathrm{inj}\ \mathrm{event}(e_3(x_1,x_2,x_3,x_4)),$$

$$\mathrm{event}(e_3(x_1,x_2,x_3,x_4))\rightsquigarrow\mathrm{inj}\ \mathrm{event}(e_2(x_1,x_2,x_3,x_4)),$$

$\mathrm{event}(e_2(x_1,x_2,x_3,x_4))\rightsquigarrow\mathrm{inj}\ \mathrm{event}(e_1(x_1,x_2,x_3))$，因为要使式(4-1)成立，为证明$e_2$被执行，甚至需要知道$e_B$被执行，而不仅仅是$e_3$已被执行，类似地，为证明$e_1$已被执行，同样需要知道$e_B$被执行，而不仅仅是$e_2$已被执行。

定义 4.9 闭进程P_0对于 Init 敌手来说满足对应性

$$\mathrm{event}(M)\Rightarrow\bigvee_{j=1}^{m}\left(\mathrm{event}(M_j)\rightsquigarrow\bigwedge_{k=1}^{l_j}[\mathrm{inj}]_{jk}q_{jk}\right)$$

其中

$$q_{\overline{jk}}=\mathrm{event}(M_{\overline{jk}})\rightsquigarrow\bigvee_{j=1}^{M_{\overline{jk}}}\bigwedge_{k=1}^{l_{\overline{jk}j}}[\mathrm{inj}]_{\overline{jk}jk}q_{\overline{jk}jk}$$

当且仅当，对任意 Init 敌手Q，对任意包含$fn(P_0)\cup\mathrm{Init}\cup fn(M)\cup\bigcup_j fn(M_j)\cup\bigcup_{\overline{jk}}fn(M_{\overline{jk}})$的$E_0$，对任意迹$\mathcal{T}=E_0,\{P_0,Q\}\rightarrow^{*}E',\mathcal{P}'$，对每一个非空的$\overline{jk}$存在函数$\phi_{\overline{jk}}$，使得对所有非空的$\overline{jk}$，$\phi_{\overline{jk}}$将$\mathcal{T}$的步骤集的一个子集映射到$\mathcal{T}$的步骤集，且有：

(1) 对所有的 τ，如果存在 σ，使得 event(σM)在$\mathcal{T}$的第 τ 步被执行，那么，存在σ'和 $J=(j_{\overline{k}})_{\overline{k}}$，使得 $\sigma' M_{j\in}=\sigma M$，且对所有非空的 $\overline{k}\phi_{\text{makejk}(\overline{k},J)}(\tau)$ 有定义，且 event$(\sigma' M_{\text{makejk}(\overline{k},J)}$在$\mathcal{T}$的第 $\phi_{\text{makejk}(\overline{k},J)}(\tau)$步被执行。

(2) 对所有非空的$\overline{jk}$，如果$[\text{inj}]_{\overline{jk}}=\text{inj}$，则 $\phi_{\overline{jk}}$是单射的。

(3) 对所有非空的$\overline{jk}$，对所有 j 和 k，如果 $\phi_{\overline{jk}jk}(\tau)$有定义，那么 $\phi_{\overline{jk}}(\tau)$有定义，且 $\phi_{\overline{jk}jk}(\tau)\leqslant\phi_{\overline{jk}}(\tau)$。对所有 j 和 k，$\phi_{jk}(\tau)$有定义，那么 $\phi_{jk}(\tau)$ $b\leqslant\tau$。

定义 4.9 中的第一项保证了要求的事件已经被执行，第二项意味着当存在 inj 标记时，对应性是单射的。第三项保证了事件以期望的顺序执行。

4.4.3 自动执行：从保密性到对应性

首先，概述一下保密性分析。子句使用两个谓词：attacker 和 message，子句通过以下方式将使用这些谓词的原子联系起来。当进程分别通过通道 $M_1,\cdots,M_n$ 收到 $M'_1,\cdots,M'_n$后，再通过通道 M 输出 M'，可生成子句 message$(M_1,M'_1)\wedge\cdots\wedge$ message$(M_n,M'_n)\Rightarrow$ message(M,M')。当攻击者可从 $M_1,\cdots,M_n$ 计算出 M 时，可生成子句 attacker$(M_1)\wedge\cdots\wedge$ attacker$(M_n)\Rightarrow$attacker(M)。子句 message$(x,y)\wedge$attacker$(x)\Rightarrow$attacker(y)表示如果攻击者拥有通道 x，则他可在 x 上实施监听。子句attacker$(x)\wedge$attacker$(y)\Rightarrow$message(x,y)表示攻击者可在他所拥有的任何信息上发送他所拥有的任何消息。当 attacker(M)可从子句中派生出时，攻击者可能拥有 M，也就是，当从子句中不能派生出 attacker(M)时，可以肯定攻击者不拥有 M，反之不然。因为 Horn 子句可以被任意次地应用，一般来说对进程的所有行为来说，并不是都为真。类似地，当 message(M,M')可由子句派生时，消息 M'也许在通道M 上发送过。因此，我们的分析夸大了行为的执行。

接下来关注对对应性的证明，如 event$(e_1(x))\rightsquigarrow$event$(e_2(x))$，为了证明该对应性，可以夸大事件 e_1 的执行：如果在此夸大的情况下可证明对应性，那么从严格意义上来说，它也是成立的。这样通过附加一个谓词 event 可以很容易地扩展对保密性的分析，这时，event(M)意味着 event(M)可能已经发生。当进程分别通过通道 $M_1,\cdots,M_n$ 收到 $M'_1,\cdots,M'_n$后执行事件 event(M)，可生成子句 message$(M_1,M'_1)\wedge\cdots\wedge$message$(M_n,M'_n)\Rightarrow$event$(M)$。然而这一夸大不能被用于 e_2：如果夸大了 e_2 的执行再来证明对应性，那么就不能真正地确定 e_2 将被执行，这样在严格意义上，对应性就会出错。所以必须用不同的方法来对待 e_2。

采用的思想如下：固定允许执行的事件集 $e_2(M)$为$\mathcal{E}$，在证明 event$(e_1(x))\rightsquigarrow$event$(e_2(x))$时，检查只有针对 M 的事件 $e_1(M)$能够使得 $e_2(M)\in\mathcal{E}$被执行。如果对$\mathcal{E}$中的任意值都证明了这一属性，那么就证明了期望的对应性。为此，引入一个谓词 m-event，使得 m-event$(e_2(M))$为真，当且仅当 $e_2(M)\in\mathcal{E}$。当进程在执行了事件 $e_2(M_0)$并分别通过通道 $M_1,\cdots,M_n$ 收到$M'_1,\cdots,M'_n$后，将 M'通过通道 M 输出，可生成子句 message$(M_1,M'_1)\wedge\cdots\wedge$message$(M_n,M'_n)\wedge m$-event$(e_2(M_0))\Rightarrow$message$(M,M')$，换句话说，只有在 m-event$(e_2(M_0))$为真，即$e_2(M_0)\in\mathcal{E}$时，在通道 M 输出 M'的行为才能执行。

例如，如果事件 $e_2(M_1)$和 $e_2(M_2)$在协议的某个迹中被执行，定义 $\mathcal{E}=\{e_2(M_1),e_2(M_2)\}$，使得 m-event$(e_2(M_1))$和 m-event$(e_2(M_2))$为真，而其他的 m-event 事实为假。然后，表明对于事件 e_1 来说，可能被执行的只有 $e_1(M_1)$和 $e_1(M_2)$。对$\mathcal{E}$中的所有值证明得出类似的结果，从而也证明了期望的对应性。

为了确定一个原子是否能够从子句中派生，使用基于归结(Resolution-based)的算法，归结执行的目标是$\mathcal{E}$中一个未知值。基本上，会不加评估地保持 m-event 原子(由于$\mathcal{E}$未知，故无法评估)。在归结的词汇表中，从不选择 m-event 原子，从而所获得的结果对$\mathcal{E}$中的任意值都成立，使得对应性得以证明。为了证明对应性 event($e_1(x)$)$\rightsquigarrow$event($e_2(x)$)，需要表明只有 m-event($e_2(M)$)成立时 event($e_1(M)$)才是可派生的。将最初的子句集转换为可派生同一原子的子句集。在所得到的子句集中，如果能够得到结论 event($e_1(M)$)的子句的前提中都包含有 m-event($e_2(M)$)，那么，只有当 m-event($e_2(M)$)成立时，event($e_1(M)$)才是可派生的，从而期望的对应性成立。

还有个问题需要解决。为简单起见，将表示消息的项直接在子句中使用。然而，为了描述保密性分析中的 nonce，使用一种特殊编码的名字 a，它由限制(νa)创建，并使用一个函数 $a[M_1,\cdots,M_n]$来描述，其中，$M_1,\cdots,M_n$ 为在限制(νa)之上收到的消息，从而，在分析时收到不同消息后创建的名字将是可区分的(这一点对分析的准确性来说很重要)。然而，在这种编码方式下，当收到相同的消息后进行相同的限制所产生的名字仍然会被合并。为解决这一问题，要给每一个重复加一个会话标识 i，i 为一整数，它对由重复所生成的每个进程副本取不同的值。将这一会话标识作为参数加入到名字的编码中形成 $a[M_1,\cdots,M_n,i_1,\cdots,i_{n'}]$，其中 $i_1,\cdots,i_{n'}$ 为限制(νa)之前的重复的会话标识。例如，在进程!$c(x)(\nu a)$中，由(νa)创建的名字被描述为 $a[x,i]$，每一次重复的执行都关联了一个不同的会话标识，从而每个名字都有不同的编码。

4.4.4　从进程到 Horn 子句

1. 标识进程(Instrumented Processes)

用闭进程 P_0 表示要检查的协议，在检查前先要通过改名，使得初始时，各进程、属性、环境，以及敌手的初始知识等描述中的不同变量所使用的标识符各不相同。对名字进行编码后，将用模式 p 来描述项，其语法如表 4.5 所示。

表 4.5　模式 p 描述项的语法

p ::=	模　式
x,y,z,i	变量
$a[p_1,\cdots,p_n,i_1,\cdots,i_{n'}]$	名字
$f(p_1,\cdots,p_n)$	构造子应用

$a[\cdots]$中的 a 被称为名称函数符(这里 a 用括号而不用加圆括号，只是为了与构造子相区别)。如果 a 是一个自由名，那么其编码可简单地记为 $a[]$。为了形式化地定义与名字相关联的模式，使用标识的概念。标识进程的语法定义如表 4.6 所示。

表 4.6　标识进程的语法定义

P,Q ::=	标识进程
$!^i P$	重复
$(\nu a:\ell)P$	限制
…(同标准演算)	

重复!P 中被加入一个变量 i，用以标识 P 的哪一个副本(即哪个会话)被执行。

限制$(\nu a)P$被加了一个限制标签l,变成$(\nu a:\ell)P$,其中l可取两种形式:在诚实进程的限制中取$a[M_1,\cdots,M_n,i_1,\cdots,i_{n'}]$,在敌手进程的限制中取$b_0[a[i_1,\cdots,i_{n'}]]$,$b_0$是一个用于敌手的特定的名字函数符。在此$l$可以看做是$a$的类型(由于包含变量,所以是多态类型)。

对于标识进程来说,语义格局用S、E、$\mathcal{P}$表示,其中S表示未被$\mathcal{P}$使用过的会话标识符集合。E为环境,它将名字映射到$a[\cdots]$形式的闭模式。$\mathcal{P}$为标识进程的有限多重集。第一个语义格局用会话标识符的任意可数集S_0。E的域必须包括$\mathcal{P}$中进程所包含的所有自由名字,初始环境将所有名字a映射为模式$a[]$。语义规则(Red Repl)和(Red Res)变为:

$$S,E,\mathcal{P}\cup\{!^iP\}\rightarrow S\backslash\{\lambda\},E,\mathcal{P}\cup\{P\{\lambda/i\},!^iP\}\ \text{where}\ \lambda\in S \qquad \text{(Red Repl)}$$

$$S,E,\mathcal{P}\cup\{(\nu a:\ell)P\}$$

$$\rightarrow S,E[a'\rightarrow E(\ell)],\mathcal{P}\cup\{P\{a'/a\}\}\ \text{if}\ a'\notin \text{dom}(E) \qquad \text{(Red Res)}$$

其他的语义规则$E,\mathcal{P}\rightarrow E,\mathcal{P}'$简单地变为$S,E,\mathcal{P}\rightarrow S,E,\mathcal{P}'$。

根据以上关于标识进程的语法可以将进程P_0转换为标识进程$P_0'=\text{instr}(P_0)$,从而使得不同的名字不会被验证器合并。

对于敌手进程Q来说,其标记进程记为$Q'=\text{instrAdv}(Q)$。给Q的每个限制(νa)加标签$b_0[a[s]]$,变成$(\nu a:b_0[a[s]])$,其中s为进程标识符序列。在此并没有将以前所收到的消息变量作为a的参数,因为对于Init敌手来说,并没有确定的关于攻击者所生成的nonce与其所收到消息之间有何联系的信息。

注意:通过把一个限制从进程的语法树中自上而下地移动,当限制移到输入、重复、析构子应用之下时,可对用以描述新鲜名字的模式加入更多的参数。可见,该转换可让分析更精确。这种转换可由工具自动地完成。

例 4.2　由4.4.1小节中的进程生成的标识进程:

$$P_A'(sk_A,pk_A,pk_B)=!_C^{i_A}(x_pk_B).(\nu a:a[x_pk_B,i_A])\ldots(\nu r_1:r_1[x_pk_B,i_A])\ldots$$
$$c(m)\ldots(\nu r_3:r_3[x_pk_B,m,i_A]])$$

$$P_B'(sk_B,pk_B,pk_A)=!_C^{i_B}(m')\ldots(\nu b:b[m',i_B])\ldots(\nu r_2:r_2[m',i_B])\ldots$$

$$P'=(\nu sk_A:sk_A[])(\nu sk_B:sk_B[])\ldots(P_A'(sk_A,pk_A,pk_B)\mid P_B'(sk_B,pk_B,pk_A))$$

标识进程的语义允许它执行和其相应标准进程相同的通信和事件。

下面来定义标识进程的对应性。这些对应性和子句使用了由表4.7所示语法定义的事实。

表 4.7　对应性和子句使用的语法定义的事实

$F::=$	事　实
attacker(p)	攻击者知识
message(p,p')	通道上的消息
m-event(p)	必须事件
event(p)	可能事件

事实attacker(p)表示攻击者可能拥有p,事实message(p,p')表示消息p'可能出现在通道p上。事实m-event(p)表示事件event(M)必须已经被执行过(M对应于p)。event(p)表示事件event(M)可能已经被执行过(M对应于p)。对应性中不使用m-event(p),但在子句中会用到。

定义 4.10 令 P_0 为一闭进程，且 $P_0'=\text{instr}(p_0)$，标记进程 P_0'对于 Init 敌手来说满足对应性

$$F\Rightarrow\bigvee_{j=1}^{m}\left(F_j\rightsquigarrow\bigwedge_{k=1}^{l_j}\text{event}(p_{jk})\right)$$

当且仅当对任意 Init 敌手 Q，对任意迹$\mathcal{T}=S_0,E_0,\{P_0',Q'\}\rightarrow^* S',E',\mathcal{P}'$，其中，$Q'=\text{instrAdv}(Q)$，对任意 $a\in\text{dom}(E_0)$有 $E_0(a)=a[\,]$，且 $f_n(P_0')\cup\text{Init}\subseteq\text{dom}(E_0)$，如果存在替换 σ，使得$\mathcal{T}$满足 σF，那么存在 σ'及 $j\in\{1,\cdots,m\}$使得 $\sigma'F_j=\sigma F$，且对所有 $k\in\{1,\cdots,l_j\}$，$\mathcal{T}$满足事件($\sigma' p_{jk}$)。

引理 4.1 标记进程的对应性蕴涵标准进程的对应性。

2. Horn 子句的生成

给定一个闭进程 P_0 及一个名字的集合 Init，协议验证器首先标记 P_0 以获得 $P_0'=\text{instr}(P_0)$，然后生成一 Horn 子句集描述协议与任意 Init 敌手的并行。子句常采用 $F_1\wedge\cdots\wedge F_n\Rightarrow F$ 的形式，其中，$F_1,\cdots,F_n,F$ 为事实，它们组成了攻击者子句和协议子句。这些子句形成集合$\mathcal{R}_{P_0',\text{Init}}$。谓词 m-event 由一个闭事实的集合$\mathcal{F}_{\text{me}}$定义，使得 m-event(p)为真当且仅当 m-event(p)$\in\mathcal{F}_{\text{me}}$。$\mathcal{F}_{\text{me}}$中的事实不属于$\mathcal{R}_{P_0',\text{Init}}$，$\mathcal{F}_{\text{me}}$是事实的集合，对应于所允许的事件集$\mathcal{E}$。以下分别从攻击者的角度和协议描述本身的角度介绍 Horn 子句的生成。

首先，反映攻击者能力的子句可通过以下方式生成。

对任一 $a\in\text{Init}$，有 attacker($a[\,]$) (Init)

attacker($b_0[x]$) (Rn)

对任一公有的 n 元构造子 f ，有

$\text{attacker}(x_1)\wedge\cdots\wedge\text{attacker}(x_n)\Rightarrow\text{attacker}(f(x_1,\cdots,x_n))$ (Rf)

对任一公有析造子 g，对任一重写规则 $g(M_1,\cdots,M_n)\rightarrow M$ in def(g)，有

$\text{attacker}(M_1)\wedge\cdots\wedge\text{attacker}(M_n)\Rightarrow\text{attacker}(M)$ (Rg)

$\text{message}(x,y)\wedge\text{attacker}(x)\Rightarrow\text{attacker}(y)$ (Rl)

$\text{attacker}(x)\wedge\text{attacker}(y)\Rightarrow\text{message}(x,y)$ (Rs)

如果 $c\in\text{Init}$，可以将子句中所有 message($c[\,],M$)的出现用 attacker(M)来代替。

其次，介绍针对协议的子句。

当一个函数 ρ 将模式与每个名字和变量关联起来，f 为构造子，通过下式将 ρ 扩展为一个替换 $\rho(f(M_1,\cdots,M_n))=f(\rho(M_1),\cdots,\rho(M_n))$。

进程 P 的转换$[\![P]\!]\rho H$ 是一个子句集，其中 H 是 message(p,p')或 m-event(p)形式的事实序列，环境 ρ 将每个变量和名字映射到其模式描述。H 序列用于跟踪已执行的事件及进程已收到消息，因为这些可能触发其他消息。ϕ 表示空序列。$H\wedge F$ 表示将事实 F 串联到序列 H，模式 ρi 为一进程标识变量。

$$[\![0]\!]\rho H=\phi$$
$$[\![P\mid Q]\!]\rho H=[\![P]\!]\rho H\cup[\![Q]\!]\rho H$$
$$[\![!^i P]\!]\rho H=[\![P]\!](\rho[i\mapsto i])H$$
$$[\![(\nu a:a[M_1,\cdots,M_n,i_1,\cdots,i_{n'}])P]\!]\rho H=$$
$$[\![P]\!](\rho[a\mapsto a[\rho(M_1),\cdots,\rho(M_n),\rho(i_1),\cdots,\rho(i_{n'})]])H$$

$[\![M(x).P]\!]\rho H = [\![P]\!](\rho[x \mapsto x])(H \wedge \text{message}(\rho(M), x))$

$[\![\overline{M}\langle N\rangle.P]\!]\rho H = [\![P]\!]\rho H \cup \{H \Rightarrow \text{message}(\rho(M), \rho(N))\}$

$[\![\text{let } x = g(M_1, \cdots, M_n) \text{ in } P \text{ else } Q]\!]\rho H = \cup \{[\![P]\!]((\sigma\rho)[x \mapsto \sigma' p'])(\sigma H)$

$g(p'_1, \cdots, p'_n) \rightarrow p'$ 在 def(g) 中，且(σ, σ') 为使 $\sigma\rho(M_1) = \sigma' p'_1, \cdots,$

$\sigma\rho(M_n) = \sigma' p'_n$的最一般替换对 $\cup [\![Q]\!]\rho H$

$[\![\text{if } M = N \text{ then } P \text{ else } Q]\!]\rho H = [\![P]\!](\sigma\rho)(\sigma H) \cup [\![Q]\!]\rho H$

其中，σ 为 $\rho(M)$和 $\rho(N)$的最一般合一(Most General Unifier, MGU)

$$[\![\text{event}(M).P]\!]\rho H = [\![P]\!]\rho(H \wedge m\text{-event}(\rho M))) \cup \{H \Rightarrow \text{event}(\rho M))\}$$

为简单起见，对析构应用的转换中，假设析构子的 else 分支总会执行，在多数情况下这种假设是多余的，因为 else 分支经常为空或仅发送错误信息。

对事件的转换将前提 m-event$(\rho(M))$加入到前提中，表明只有在该事件首先被执行了的情况下才执行进程 P。另外，还加入了一个子句，意思是，当 H 中的所有条件为真时便触发了该事件。

注意：可以证明，对不同形式的对应性，有时可以简化事件子句的生成。假设进程和对应性中所有事件的参数为 $f(M_1, \cdots, M_n)$形式的。如果事件集event$(f(\cdots))$在期望的对应性中仅出现在$\rightsquigarrow$之前，那么子句 m-event$(f(\cdots))$形式的前提可被去掉而不会改变结果，这时，由事件 event(M)生成的子句可被简化为

$$[\![\text{event}(M).P]\!]\rho H = [\![P]\!]\rho H \cup \{H \Rightarrow \text{event}(\rho(M))\}$$

(直观地说，由于 event$(f(\cdots))$在期望的对应性中仅出现在$\rightsquigarrow$之前，我们根本不会去证明 event$(f(\cdots))$已经被执行过，这时，事实 m-event$(f(\cdots))$是无用的。)

类似地，如果 event$(f(\cdots))$在期望的对应性中仅出现在$\rightsquigarrow$之后，那么结论为 event$(f(\cdots))$形式的子句可被去掉而不会改变结果。这时，由事件 event(M)生成的子句可被简化为

$$[\![\text{event}(M).P]\!]\rho H = [\![P]\!]\rho(H \wedge m\text{-event}(\rho(M)))$$

(直观地说，由于事件集 event$(f(\cdots))$在期望的对应性中仅出现在$\rightsquigarrow$之后，我们不会去证明以下形式的属性“如果 event$(f(\cdots))$已经被执行，那么…”，所以结论为 event$(f(\cdots))$的子句是无用的。)

这种从协议到 Horn 子句的转换引入了近似。由于子句可被任意次地应用，所以行为被隐式地重复了。对于那种要求某值保密，而后又公开它的协议来说，这种近似蕴涵了工具将不能用来证明这种协议。例如，考虑以下进程$(\nu d)(\overline{d}\langle s\rangle.\overline{c}\langle d\rangle \mid d(x))$，其中在 d 被输出到公有通道 c 上，从而敌手知道了 d 之前 s 是保密的，因为这时 d 是私有的，而 s 的输入与输出都是通过该私有通道进行的。然而，Horn 子句方法不能证明这一属性，因为它将进程看成以下附加了重复的变体$(\nu d)(!\overline{d}\langle s\rangle.\overline{c}\langle d\rangle \mid !d(x))$，使其不能反映 s 在敌手知道 d 之前的保密性。类似地，进程$(\nu d)(\overline{d}\langle M\rangle \mid d(x).d(x).\text{event}(e_1))$永远不会执行事件 e_1，但 Horn 子句方法能证明这一属性，因为它将该进程当成了$(\nu d)(!\overline{d}\langle M\rangle \mid d(x).d(x).\text{event}(e_1))$，使得 e_1 的执行成为可能。由于这种近似，工具是不完备的(Complete)(它可能生成错误的攻击)，但这表明它是可靠的(Sound)(它所证明的属性总是为真)。

最后，将以上生成的子句汇总起来便可得到验证中所需的子句。

令 $\rho=\{a\mapsto a[\,]\mid a\in f_n(P'_0)\}$，定义相应于标记进程 P' 的子句如下：

$$\mathcal{R}_{P'_0,\mathrm{Init}}=[\![P'_0]\!]\rho\phi\ \cup\ \{\mathrm{attacker}(a[\,])\mid a\in \mathrm{Init}\}$$
$$\cup\ \{(\mathrm{Rn}),(\mathrm{Rf}),(\mathrm{Rg}),(\mathrm{Rl}),(\mathrm{Rs})\}$$

例 4.3 4.4.1 小节进程 P 的子句为敌手子句加上以下子句：

$attacker(\mathrm{p}k(sk_A[\,]))$ (4-2)

$\mathrm{attacker}(pk(sk_B[\,]))$ (4-3)

$H_1\Rightarrow\mathrm{attacker}(\mathrm{pencrypt_p}((a[x_pk_B,i_A],pk(sk_A[\,])),x_pk_B,r_1[x_pk_B,i_A]))$ (4-4)

$H_2\Rightarrow\mathrm{attacker}(\mathrm{pencrypt_p}(x_b,x_pk_B,r_3[x_pk_B,p_2,i_A]))$ (4-5)

$H_3\Rightarrow\mathrm{event}(e_A(pk(sk_A[\,]),pk(sk_B[\,]),a[pk(sk_B[\,]),i_A],x_b))$ (4-6)

$H_3\Rightarrow\mathrm{attacker}(\mathrm{sencrypt}(sAa[\,],a[pk(sk_B[\,]),i_A]))$ (4-7)

$H_3\Rightarrow\mathrm{attacker}(\mathrm{sencrypt}(sAb[\,],x_b))$ (4-8)

where $p2=\mathrm{pencrypt_p}((a[x_pk_B,i_A],x_b,x_pk_B),pk(sk_A[\,]),x_r_2)$

$H_1=\mathrm{attacker}(x_pk_B)\wedge m\text{-}\mathrm{event}(e_1(pk(sk_A[\,]),x_pk_B,a[x_pk_B,i_A]))$

$H_2=H_1\wedge\mathrm{attacker}(p_2)\wedge m\text{-}\mathrm{event}(e_3(pk(sk_A[\,]),x_pk_B,a[x_pk_B,i_A],x_b))$

$H_3=H_2\{pk(sk_B[\,])/x_pk_B\}$

$\mathrm{attacker}(p_1)\wedge m\text{-}\mathrm{event}(e_2(x_pk_A,pk(sk_B[\,]),x_a,b[p_1,i_B]))$

$\Rightarrow\mathrm{attacker}(\mathrm{pencrypt_p}((x_a,b[p_1,i_B],pk(sk_B[\,])),x_pk_A,r_2[p_1,i_B]))$ (4-9)

where $p_1=\mathrm{pencrypt_p}((x_a,x_pk_A),pk(sk_B[\,]),x_r_1)$

$H_4\Rightarrow\mathrm{event}(e_B(pk(sk_A[\,]),pk(sk_B[\,]),x_a,b[p'_1,i_B]))$ (4-10)

$H_4\Rightarrow\mathrm{attacker}(\mathrm{sencrypt}(sBa[\,],x_a))$ (4-11)

$H_4\Rightarrow\mathrm{attacker}(\mathrm{sencrypt}(sBb[\,],b[p'_1,i_B]))$ (4-12)

where $p'_1=\mathrm{pencrypt_p}((x_a,pk(sk_A[\,])),pk(sk_B[\,]),x_r_1)$

$H_4=\mathrm{attacker}(p'_1)\wedge m\text{-}\mathrm{event}(e_2(pk(sk_A[\,]),pk(sk_B[\,]),x_a,b[p'_1,i_B]))\wedge$
$\quad\mathrm{attacker}(\mathrm{pencrypt_p}(b[p'_1,i_B],pk(sk_B[\,]),x_r_3))$

定理 4.1(子句的正确性) 设 P_0 为一闭进程，Q 为一 Init 敌手。令 $P'_0=\mathrm{instr}(P_0)$，$Q'=\mathrm{instrAdv}(Q)$，迹 $\mathcal{T}=S_0,E_0,\{P'_0,Q'\}\rightarrow^* S',E',\mathcal{P}'$，且有 $f_n(P'_0)\cup\mathrm{Init}\subseteq\mathrm{dom}(E_0)$，对所有的 $a\in\mathrm{dom}(E_0)$，$E_0(a)=a[\,]$，假设：如果 $\mathcal{T}$ 满足 $\mathrm{event}(p)$，则 $m\text{-}\mathrm{event}(p)\in\mathcal{F}_{me}$，并假设 $\mathcal{T}$ 满足 F，那么，F 可从 $\mathcal{R}_{P'_0,\mathrm{Init}}\cup\mathcal{F}_{me}$ 中派生出来。

4.4.5 求解算法

1. 基本算法

为了应用前面的结果，必须决定一个事实能否从 $\mathcal{R}_{P'_0,\mathrm{Init}}\cup\mathcal{F}_{me}$ 中派生出来。这也许是不可判定的，但在实际中存在对大量例子协议可终止的算法。特别地，可使用归结算法的变体。首先定义归结：

定义 4.11 设 $R=H\Rightarrow C$，$R'=H'\Rightarrow C'$ 为两个子句，假设存在 $F_0\in H'$ 使得 C 和 F_0 是可合一的，σ 为 C 和 F_0 其最一般合一(MGU)，这时定义 $R\circ_{F_0}R'=\sigma(H\cup(H'\setminus\{F_0\}))\Rightarrow\sigma C'$。

获得高效求解算法的一种重要思想是在保持完备性的同时指定限制归结的应用条件。用在归结上的条件相当于自由选择的归结:选择函数在每个子句中挑出被选出的事实,归结只被应用在这些被选择的事实,也就是说,只有当结论在 R 中被选择,F_0 在 R'中被选择时才会生成子句 $R\circ_{F_0}R'$。

定义 4.12 用 sel 表示一个选择函数,即从子句到事实集的函数,使得 $\mathrm{sel}(H\Rightarrow C)\subseteq H$。如果 $F\in\mathrm{sel}(R)$,则称 F 在 R 中被选择。如果 $\mathrm{sel}(R)=\phi$,则称 R 中的前提都未被选择,或者子句的结论被选择。

选择函数的选择可显著地加速算法。由于只有当在归结中合一的事实被选择时才会引起算法根据归结法对子句进行组合,所以挑选选择函数以减少被选事实间可能的合一数量。存在被选事实会减慢算法,因为有更多的归结选择可执行,因此,在每个子句中最多选择一个事实。对协议来说,有 x 变量的 attacker(x)形式的事实可以和所有有 attacker(p)形式的事实合一,所以要避免选择它们。m-event 事实永远不会被选择,因为它们不会被已知的子句所定义。

定义 4.13 如果存在变量 x 使得事实 $F=\mathrm{attacker}(x)$,或存在模式 p,使得 $F=m\text{-}\mathrm{event}(p)$,则称事实 F 是不可选择的(Unselectable),否则称 F 是可选择的(Selectable)。

安全协议的一个基本的选择函数为

$$\mathrm{sel}_0(H\Rightarrow C)=\begin{cases}\phi & \text{如果对任意的} F\in H, F \text{是不可选择的}\\ \{F_0\} & F_0\in H \text{ 且 } F_0 \text{ 是可选择的,否则}\end{cases}$$

在实现中,前提被表示为一个列表,被选择的事实是前提列表中第一个可选的元素。

求解算法的过程分为两个阶段,见表 4.8。第一个阶段为饱和(Saturate),将子句集转换到一个等价但更简单的子句集。第二个阶段为派生(Derivable),采用先深搜索的方法以判定一个事实能否由子句导出。

表 4.8 求解算法

第一阶段:饱和
saturate($\mathcal{R}_0$)=
1. $\mathcal{R}\leftarrow\phi$.
For each $R\in\mathcal{R}_0$, $\mathcal{R}\leftarrow\mathrm{elim}(\mathrm{simplify}(R)\cup\mathcal{R})$.
2. Repeat until a fixpoint is reached
for each $R\in\mathcal{R}$ such that $\mathrm{sel}(R)=\phi$,
for each $R'\in\mathcal{R}$, for each $F_0\in\mathrm{sel}(R')$ such that $R\circ_{F_0}R'$ is defined,
$\mathcal{R}\leftarrow\mathrm{elim}(\mathrm{simplify}(R\circ_{F_0}R')\cup\mathcal{R})$.
3. Return $\{R\in\mathcal{R}\mid \mathrm{sel}(R)=\phi\}$.
第二阶段:反向先深搜索
$\mathrm{deriv}(R,\mathcal{R},\mathcal{R}_1)=\begin{cases}\phi & \text{if } \exists R'\in\mathcal{R}, R'\sqsupseteq R\\ \{R\} & \text{otherwise, if } \mathrm{sel}(R)=\phi\\ \bigcup\{\mathrm{deriv}(\mathrm{simplify}'(R'\circ_{F_0}R),\{R\}\cup\mathcal{R},\mathcal{R}_1)\mid R'\in\mathcal{R}_1,\\ \quad F_0\in\mathrm{sel}(R) \text{ such that } R'\circ_{F_0}R \text{ is defined}\} & \text{otherwise}\end{cases}$
$\mathrm{derivable}(F,\mathcal{R}_1)=\mathrm{deriv}(F\Rightarrow F,\phi,\mathcal{R}_1)$

表 4.8 中$\mathcal{R}_0$表示最初用以描述协议和攻击者的子句集,simplify 为一化简函数,elim 函数用以消除被包含子句。称 $H_1\Rightarrow C_1$ 包含 $H_2\Rightarrow C_2$ 是指,存在一个替换 σ,使得 $\sigma C_1=C_2$ 且

$\sigma H_1 \subseteq H_2$。记为$(H_1 \Rightarrow C_1) \sqsupseteq (H_2 \Rightarrow C_2)$。本质上，饱和保持了派生性。

派生阶段搜索可从$\mathcal{R}_1 = \text{saturate}(\mathcal{R}_0)$中导出的事实。derivable$(F, \mathcal{R}_1)$返回一个空选择的子句集 $R = H \Rightarrow C$，使得 R 可通过归结从$\mathcal{R}_1$得到。C 为 F 的实例，且所有可从$\mathcal{R}_1$派生出的 F 的实例都可以通过将 derivable$(F, \mathcal{R}_1)$中的一个子句作为最后一个子句而派生出来。

搜索本身是由 deriv$(R, \mathcal{R}, \mathcal{R}_1)$执行的。函数 deriv 从 $R = F \Rightarrow F$ 开始，并通过使用$\mathcal{R}_1$中的子句 R'对 R 的前提进行变形，以派生 R 前提中的一个元素 F_0，这样 R 就被 $R' \circ_{F_0} R$ 代替(deriv 定义中的第三种情况)。F_0 的选择使用选择函数 sel。集合$\mathcal{R}$为在搜索过程中已经看到过的子句的集合。最初$\mathcal{R}$为空，子句 R 在 deriv 定义的第三种情形下被加入到$\mathcal{R}$中。

上述对 R 的变形一直进行，直到下列两个条件满足：

(1) R 被$\mathcal{R}$中的子句包含：这时进入循环，正在找的是以前已经找过的事实的实例。

(2) sel(R)为空：得到适用子句 R 并返回它。

2. 简化步骤

(1) 数据构造子的分解。函数 decomp 将 attacker$(f(p_1, \cdots, p_n))$形式的事实替换为 attacker$(p_1) \wedge \cdots \wedge$ attacker(p_n)，当这一替换用于子句 $H \Rightarrow$attacker$(f(p_1, \cdots, p_n))$时，就会产生 n 个如下形式的子句：$H \Rightarrow$attacker(p_i)，$i \in \{1, \cdots, n\}$。函数 decomphyp 仅在子句的前提中进行这种分解。

(2) 消除同义反复。函数 elimtaut 将那些结论已经在前提中的子句去掉。

(3) 消除重复前提。函数 elimdup 将子句的重复前提去掉。

(4) 消除无用的 attacker(x)前提。如果子句 $H \Rightarrow C$ 的前提中包含有attacker(x)，其中 x 为一变量，且 attacker(x)在子句的其他地方未出现，则前提中的attacker(x)将被函数 elimattx 消除掉。事实上，攻击者总会拥有至少一个消息，所以 attacker(x)总是满足的。

(5) 秘密假设。如果用户知道一个事实 F 是不可派生的，他可以将此告诉验证器。令$\mathcal{F}_{\text{not}}$为用户声明的不可派生的事实集合，函数 elimnot 会消除掉所有前提中有$\mathcal{F}_{\text{not}}$中事实实例的子句。例如，由于攻击者不能知道主体的私钥，所以可以将 attacker$(sk_A[\,])$和 attacker$(sk_B[\,])$声明为不可派生事实。

这种不可派生事实对搜索空间实施剪枝，通过消除无用子句加速了搜索进程。下面为总的求解算法。

solve$_{P'_0, \text{Init}}(F)=$

① Let $\mathcal{R}_1 = \text{saturate}(\mathcal{R}_{P'_0, \text{Init}})$

② For each $F' \in \mathcal{F}_{\text{not}}$, if derivable$(F', \mathcal{R}_1) \neq \phi$, then terminate with error

③ Return derivable$(F, \mathcal{R}_1)$

(6) 消除冗余前提。当一个子句的形式为 $H \wedge H' \Rightarrow C$，且存在 σ，使得 $\sigma H \subseteq H'$，且 σ 不改变 H'和 C 中的变量，则由函数 elimredundanthyp 将该子句替换为 $H' \Rightarrow C$。

(7) 将所有简化集中起来。

simplify=elimattx∘elimtaut∘elimnot∘elimredundanthyp∘elimdup∘decomp

simplify′=elimattx∘elimnot∘elimredundanthyp∘elimdup∘decomphyp

simplify′中使用了 decomphyp 而不是 decomp，原因在于它所考虑的子句的结论是要派

生的事实，它是不能被改变的。

3. 可靠性

引理 4.2（saturate 的正确性） 设 F 为一个闭事实。如果对所有 $F' \in \mathcal{F}_{not}$，F' 的实例均不能由 saturate$(\mathcal{R}_0) \cup \mathcal{F}_{me}$ 派生，那么 F 可由 $R_0 \cup \mathcal{F}_{me}$ 派生，当且仅当 F 可由 saturate$(\mathcal{R}_0) \cup \mathcal{F}_{me}$ 派生。

引理 4.3（derivable 的正确性） 设 F' 为 F 的一个闭实例。如果对所有的 $F'' \in \mathcal{F}_{not}$，derivable$(F'', \mathcal{R}_1) = \phi$，那么 F' 可由 $\mathcal{R}_1 \cup \mathcal{F}_{me}$ 派生，当且仅当存在一个 σ，且 $H \Rightarrow C$ 中存在一个子句，使得 $\sigma C = F'$，且 σH 中的所有元素可由 $\mathcal{R}_1 \cup \mathcal{F}_{me}$ 派生。

定理 4.2（主定理） 设 P_0 为一个闭进程，且 $P_0' = \text{instr}(P_0)$，Q 为一 Init 敌手，且 $Q' = \text{instrAdv}(Q)$。考虑一个迹 $\mathcal{T} = S_0, E_0, \{P'_0, Q'\} \rightarrow^* S', E', P'$，其中 $fn(P'_0) \cup \text{Init} \subseteq \text{dom}(E_0)$，且对所有的 $a \in \text{dom}(E_0)$ 有 $E_0(a) = a[\,]$。如果 $\mathcal{T}$ 满足 F 的一个实例 F'，那么存在一个子句 $H \Rightarrow C \in \text{solve}_{P'_0,\text{Init}}(F)$，一个替换 σ，使得 $F' = \sigma C$，且对 σH 中所有的 m-event(p)，$\mathcal{T}$ 满足 event(p)。

定理 4.2 是主要的正确性结果，它允许人们证明一些事件一定已经被执行过。

例 4.4 对 4.4.1 小节的进程 P，Init$=\{c\}$，$P' = \text{instr}(P)$，工具表明

$$
\begin{aligned}
\text{solve}_{P',\text{Init}}(\text{event}(e_B(x_1, x_2, x_3, x_4))) = \{ & m\text{-event}(e_1(pk_A, pk_B, p_a)) \wedge \\
& m\text{-event}(e_2(pk_A, pk_B, p_a, p_b)) \wedge \\
& m\text{-event}(e_3(pk_A, pk_B, p_a, p_b)) \\
& \Rightarrow \text{event}\ (e_B(pk_A, pk_B, p_a, p_b))\}
\end{aligned}
$$

其中，

$$
\begin{aligned}
& pk_A = pk(sk_A[\,]), pk_B = pk(sk_B[\,]), pa = a[pk_B, i_A] \\
& p_b = b[\text{pencrypt}_p((p_a, pk_A), pk_B, r_1[pk_B, i_A]), i_B]
\end{aligned}
$$

根据定理 4.2，如果 $\mathcal{T}$ 满足 event$(e_B(p_1, p_2, p_3, p_4))$，该事件是 event$(e_B(x_1, x_2, x_3, x_4))$的一个实例，这样，给定 solve$_{P',\text{Init}}$(event$(e_B(x_1, x_2, x_3, x_4))$)的值，则存在 σ，使得 event$(e_B(p_1, p_2, p_3, p_4)) = \sigma$event$(e_B(pk_A, pk_B, p_a, p_b))$，且 $\mathcal{T}$ 满足

$$
\begin{aligned}
& \text{event}(\sigma e_1(pk_A, pk_B, p_a)) = \text{event}(e_1(p_1, p_2, p_3)) \\
& \text{event}(\sigma e_2(pk_A, pk_B, p_a, p_b)) = \text{event}(e_2(p_1, p_2, p_3, p_4)) \\
& \text{event}(\sigma e_3(pk_A, pk_B, p_a, p_b)) = \text{event}(e_3(p_1, p_2, p_3, p_4))
\end{aligned}
$$

因此，如果事件 event$(e_B(M_1, M_2, M_3, M_4))$执行过，则事件 event$(e_1(M_1, M_2, M_3))$，event$(e_2(M_1, M_2, M_3, M_4))$及 event$(e_3(M_1, M_2, M_3, M_4))$已经被执行过。

4. 应用于对应性

标记进程的对应性可根据以下定理来检验。

定理 4.3 令 P_0 为一闭进程，且 $P'_0 = \text{instr}(P_0)$，令 p_{jk} $(j \in \{1, \cdots, m\}, k \in \{1, \cdots, l_j\})$ 为模式，令 F 和 F_j $(j \in \{1, \cdots, m\})$ 为事实。假设对所有的 $R \in \text{solve}_{P'_0,\text{Init}}(F)$，存在 $j \in \{1, \cdots, m\}$，σ' 及 H 使得 $R = H \wedge m\text{-event}(\sigma' p_{j1}) \wedge \cdots \wedge m\text{-event}(\sigma' p_{jl_j}) \Rightarrow \sigma' F_j$，则 P'_0 对于 Init 敌手来说满足对应性 $F \Rightarrow \bigvee_{j=1}^{m} \left(F_j \rightsquigarrow \bigwedge_{k=1}^{l_j} \text{event}(p_{jk})\right)$。

定理 4.4 与定理 4.3 类似，只是针对标准进程，即非标记进程。

定理 4.4 令 P_0 为一闭进程，且 $P'_0=\text{instr}(P_0)$，令 $M_{jk}(j\in\{1,\cdots,m\},k\in\{1,\cdots,l_j\})$ 为项，令 α 和 $\alpha_j(j\in\{1,\cdots,m\}$ 为原子。令 p_{jk},F,F_j 分别为将项和原子 M_{jk},α,α_j 中的名字 a 替换为模式 $a[]$ 后所得到的模式和事实。假设对 $\text{solve}_{P'_0,\text{Init}}(F)$ 中的所有子句 R，存在 $j\in\{1,\cdots,m\}$，σ 及 H 使得 $R=H\wedge m\text{-event}(\sigma' p_{j1})\wedge\cdots\wedge m\text{-event}(\sigma' p_{jl_j})\Rightarrow\sigma' F_j$，则 P_0 对于 Init 敌手来说，满足对应性 $\alpha\Rightarrow\bigvee_{j=1}^{m}(\alpha_j\rightsquigarrow\bigwedge_{k=1}^{l_j}\text{event}(M_{jk}))$。

推论 4.1（保密性） 令 P_0 为一闭进程，且 $P'_0=\text{instr}(P_0)$，令 N 为一个项，令 p 为将项 N 中的名字 a 替换为模式 $a[]$ 后所得到的模式。假设 $\text{solve}_{P',\text{Init}}(\text{attacker}(p))=\phi$，则 P_0 对于 Init 来说保持了所有 N 的实例的秘密性。

推论 4.2（非单射协定） 令 P_0 为一闭进程，且 $P'_0=\text{instr}(P_0)$。假设对任一 $R\in\text{solve}_{P'_0,\text{Init}}(\text{event}(e(x_1,\cdots,x_n)))$ 使得 $R=H\Rightarrow\text{event}(e(p_1,\cdots,p_n))$，有 $m\text{-event}(e'(p_1,\cdots,p_n))\in H$，则 P_0 对于 Init 敌手来说满足对应性 $\text{event}(e(x_1,\cdots,x_n))\rightsquigarrow\text{event}(e'(x_1,\cdots,x_n))$。

本节通过利用逻辑编程技术，扩展了前人在安全协议验证方面的工作。该技术可用于验证各种安全属性：从保密性到通用的对应性，不仅包括认证性，而且包括可表达消息以期望的顺序被发送或接收的对应性。这一技术可以以完全自动的方式来检查对应性，且不限制协议的会话次数。在算法的终止性方面，可以证明，对于一种特定的附加标号的协议来说，算法是终止的。利用该工具对一些现有协议的验证结果表明，许多已知的协议缺陷均可通过该工具发现，且用该技术验证这些协议只需要少于 1s 的时间。

4.5 形式化方法的计算可靠性

通常，在设计、分析、实现使用密码技术的系统时，以一种抽象的视角来对待密码操作已足够了。例如，用一种高级的描述来表示加密而忽略加密函数的细节是一种很方便的做法。

对于密码操作来说，近年来至少有两种不同的抽象观点，它们都是一致的、有用的，但却截然不同，几乎可以说来自于两个互不相干的群体。其中一种观点将密码操作看成是符号化(形式化)表达式空间上的函数，其安全属性也被形式化地建模。另一种观点将密码操作看成是位串的函数，其安全性以成功攻击的概率及计算复杂性来定义。

这两种观点之间存在着一种间隙，以下将介绍如何弥合这一间隙。为了描述这两种观点，将作两种关于对称密钥的考虑：一种较简单，基于形式化系统；另一种稍复杂，基于计算模型。然后，以定理的形式给出将这两种考虑关联起来的可靠结果：用形式化方法可证明的保密性在计算模型中也保持为真。

4.5.1 形式化加密和表达式的等价

本节主要描述密码的形式化观点，给出加密操作的表达式空间以及两个表达式相等价的含义。

1. 表达式

用 Bool 表示{0,1},用 0、1 可以拼写出一些消息,如数字、主体名等。用 Keys 表示与 Bool 不相交的、固定的、非空的符号集,符号 $K, K', K'', \cdots$ 及 $K_1, K_2, \cdots$ 均在该集合中。非形式地说,Keys 中的元素表示密钥,由主体随机地生成,但形式化地说,密钥是原子符号,而不是位串。用 Exp 表示表达式的集合,它由如表 4.9 所示的语法定义。

表 4.9　用 Exp 表示表达式的集合的语法定义

$M,N:=$	表 达 式
K	密钥($K \in$ Keys)
i	位($i \in$ Bool)
(M,N)	并置
$\{M\}_K$	加密($K \in$ Keys)

其中,并置与加密可以嵌套,如表达式$(\{\{(0,K')\}_K\}_{K'},K)$。需要强调的是,Exp 中的元素是形式化的表达式,而不是实际的密钥、位、并置或加密。特别地,它们是明确的,如(M,N)等于(M',N')当且仅当 M 等于 M' 且 N 等于 N',它永远不会等于$\{M'\}_K$,类似地,$\{M\}_K$ 等于$\{M'\}_{K'}$ 当且仅当 M 等于 M' 且 K 等于 K'。然而,根据下文的定义,$\{M\}_K$ 和 $\{M'\}_{K'}$ 也许是等价的,甚至当 M 与 M' 不同,K 与 K' 不同时也会如此。

以下考虑一些对表达式的限制。如果存在 N,使得$\{N\}_K$ 为 M 的一个子表达式,且 K' 在 N 中出现,则称 K 在 M 中加密了 K'。对每一 M,以上定义了密钥上的二元关系,即"加密"关系。如果与 M 相关的"加密"关系是循环的(非循环的),则称 M 是循环的(非循环的)。如,$\{K\}_K$ 和$(\{K\}_{K'},\{K'\}_K)$都是循环的,而$(\{K\}_{K'},\{0\}_K)$是非循环的。

2. 等价关系

首先,定义一种获取关系 $M \vdash N$,其中 M 和 N 为表达式。直观地说,$M \vdash N$ 表示 N 可以从 M 中计算得出。形式化地定义该关系如下。

(1) $M \vdash 0$ and $M \vdash 1$。

(2) $M \vdash M$。

(3) if $M \vdash N_1$ and $M \vdash N_2$ then $M \vdash (N_1, N_2)$。

(4) if $M \vdash (N_1, N_2)$ then $M \vdash N_1$ and $M \vdash N_2$。

(5) if $M \vdash N$ and $M \vdash K$ then $M \vdash \{N\}_K$。

(6) if $M \vdash \{N\}_K$ and $M \vdash K$ then $M \vdash N$。

$M \vdash N$ 的定义建模了在攻击者没有关于 M 中所用到密钥 K 的先验知识的情况下,可从 M 中获取的消息。例如,有$(\{\{K_1\}_{K_2}\}_{K_3},K_3) \vdash K_3$ 及$(\{\{K_1\}_{K_2}\}_{K_3},K_3) \vdash \{K_1\}_{K_2}$,但$(\{\{K_1\}_{K_2}\}_{K_3},K_3) \vdash K_1$ 是不成立的。

从以上定义可派生出一种更一般的定义:在有关于 K 的先验知识的情况下,从 M 中获取 N,等价于在没有先验知识的情况下从(M,K)中获取 N。

其次,引入一个符号□,它表示一个攻击者不能解密的密文。定义模式的集合 Pat 作为表达式集合的扩展,其语法如表 4.10 所示。

表 4.10　集合 Pat 作为表达式集合的扩展的语法定义

$M,N::=$	模　　式
K	密钥($K\in$ Keys)
i	位($i\in$ Bool)
(M,N)	并置
$\{M\}_K$	加密($K\in$ Keys)
$\square$	不可解密

直观上，模式就是一种表达式，不过在这种表达式中可能存在有攻击者不能解密的部分。给定一个密钥集合 T 及一个表达式 M，可以通过定义以下函数将 M 转换为模式。

$$p(K,T)=K \qquad (K\in \text{Keys})$$
$$p(i,T)=i \qquad (i\in \text{Bool})$$
$$p((M,N),T)=(p(M,T),p(N,T))$$
$$p(\{M\}_K,T)=\begin{cases}\{p(M,T)\}_K & \text{若 } K\in T\\ \square & \text{其他}\end{cases}$$

直观上，该函数的结果就是当攻击者拥有 T 中的密钥时，所能够从 M 中看到的模式。

进一步地，可以定义在没有辅助集合 T 的情况下，表达式 M 的模式，这时只能使用从表达式本身可得到的密钥集合。

$$\text{pattern}(M)=p(M,\{K\in\text{Keys}\mid M\vdash K\})$$

例如，有

$$\text{pattern}((\{\{K_1\}_{K_2}\}_{K_3},K_3))=(\{\square\}_{K_3},K_3)$$

最后，来定义等价。称两个表达式是等价的，当且仅当它们生成相同的模式：

$$M\equiv N \text{ 当且仅当 } \text{pattern}(M)=\text{pattern}(N)$$

例如，有

$$(\{\{K_1\}_{K_2}\}_{K_3},K_3)\equiv(\{\{0\}_{K_1}\}_{K_3},K_3)$$

因为它们都生成模式$(\{\square\}_{K_3},K_3)$。

可以将密钥看作约束名，从而可以重命名。例如，虽然$(\{0\}_K,K)$和$(\{0\}_{K'},K')$是不等价的，但重命名后它们就是等价的了。更一般地，定义可重命名下的等价关系为$\cong$：

$$M\cong N \text{ 当且仅当存在 Keys 上的双射 } \sigma\text{，使得 } M\equiv N\sigma\text{。}$$

其中，$N\sigma$ 是将 σ 作为一个替换应用于 N 的结果。虽然$\cong$没有$\equiv$严格，但它可以顺利地用于可靠定理中，因此这里主要讲解$\cong$。在非正式情况下，不区分这两种关系，而将它们都称为等价。

3. 一些例子

(1) $0\cong 0$。

(2) $0\ncong 1$。

(3) $\{0\}_K\cong\{1\}_K$。

(4) $(K,\{0\}_K)\ncong(K,\{1\}_K)$，但$(K,\{(\{0\}_{K'},0)\}_K)\cong(K,\{(\{1\}_{K'},0)\}_K)$。

(5) $K\not\equiv K'$，但 $K\cong K'$，因为在$\cong$中密钥可以换名，而在$\equiv$中不可以。

(6) $\{0\}_K \cong \{1\}_{K'}$，而且甚至有$\{0\}_K \equiv \{1\}_{K'}$(虽然两个密文是用不同的密钥加密的)。

(7) $\{0\}_K \cong \{K\}_K$，尽管$\{K\}_K$上循环的。

(8) $\{(((1,1),(1,1)),((1,1),(1,1)))\}_K \cong \{0\}_K$。

(9) $(\{0\}_K,\{0\}_K) \cong (\{0\}_K,\{1\}_K)$。

非形式地，如果攻击者没有密钥，那么它将不能检测到两个被加密的明文是否是相同的。在实现中，这种等价可由随机加密函数来保证。

(10) $(\{0\}_K,\{1\}_K) \cong (\{0\}_K,\{1\}_{K'})$。

非形式地，如果攻击者没有密钥，那么它将不能检测到两个密文是否使用了相同的密钥。

4.5.2 计算的观点：对称加密方案及其安全性定义

本节将给出对对称加密的计算处理方法。首先，描述构造对称加密方案的函数，其次说明对称加密方案的安全性。我们所关注的安全性(称其为类型 1 安全)比通常意义上的安全性更强(如语义安全等)。然而，可以通过标准的复杂性理论假设来实现类型 1 安全。

1. 对称加密方案

设 String$=\{0,1\}^*$为所有有限串的集合，$|x|$为串 x 的长度。Plaintext、Ciphertext 及 Key 为有限串的非空集合。0 为 Plaintext 中的一个特定串。如果对一个不在 Plaintext 集合中的串进行加密形成一个密文，那么对该密文进行解密将得到 0。设 $x\in$Plaintext，则 x' 表示 Plaintext 中所有与 x 有相同长度的串。Key 被赋予了一种固定的分布(如果 Key 是有限的，那么 Key 上的分布是均匀分布)。Coins 为无穷串$\{0,1\}^\omega$，Parameter(安全参数的集合)为 1^*(由 1 组成的有限串)。

对称加密方案是由 3 个算法构成的三元组$(\mathcal{K},\mathcal{E},\mathcal{D})$，其中

$$\mathcal{K}\text{: Parameter}\times\text{Coins}\rightarrow\text{Key}$$

$$\mathcal{E}\text{: Key}\times\text{String}\times\text{Coins}\rightarrow\text{Ciphertext}$$

$$\mathcal{D}\text{: Key}\times\text{String}\rightarrow\text{Plaintext}$$

以上每个算法都是输入大小的多项式时间可计算的。$\mathcal{K}$为密钥生成算法，$\mathcal{E}$为加密算法，$\mathcal{D}$为解密算法。通常将$\mathcal{E}$或$\mathcal{D}$的第一个参数，即密钥写在下标的位置上。对所有 $\eta\in$ Parameter，$k\in\mathcal{K}(\eta)$，$r\in$ Coins，如果 $m\in$ Plaintext，则$\mathcal{D}_k(\mathcal{E}_k(m,r))=m$，但如果 $m\notin$ Plaintext，则$\mathcal{D}_k(\mathcal{E}_k(m,r))=0$。另外，当 $k\in\mathcal{K}(\eta)$时，$|\mathcal{E}_k(x)|$仅依赖于 η 和$|x|$。以上关于加密的定义是随机的、无状态的。

2. 关于加密方案的安全性

首先来看一个安全加密方案(所具备或不具备)的属性。这里选出加密方案的 3 种特征，第一种和第三种是广为人知的，第二种似乎还没有受到关注。

(1) 重复隐藏与重复泄露

给定密文 c 和 c'，能否判断出其相应的明文是相等的？如果可以，则称该方案是重复泄露的，否则是重复隐藏的。一个重复隐藏方案必须是随机的(或者有状态的)。

(2) 相同密钥隐藏和相同密钥泄露

如果有各种密钥对消息进行加密，能否判断出哪些消息是用同一密钥加密的，如果可

以，则称该方案是相同密钥泄露的，否则称其是相同密钥隐藏的。

(3) 消息长度隐藏和消息长度泄露

密文是否泄露了其背后明文的长度？如果是，则称该方案是消息长度泄露的，否则称其是消息长度隐藏的。

这 3 种特征是正交的，它可形成 8 种组合，分别将这 8 种安全概念称为类型 0、类型 1、类型 2、……、类型 7。其中的数字以以下方式决定：隐藏对应于 0，泄露对应于 1，将以上 3 种特征解释为 3 位的二进制数字，第 1 位表示重复隐藏或重复泄露，第 2 位表示相同密钥隐藏或相同密钥泄露，第 3 位表示消息长度隐藏或消息长度泄露。根据这种表示，通常所用的安全性一般是类型 3 安全的，即密文可泄露消息长度以及使用了哪个密钥，但它不能泄露两个密文是否是对同一个明文的加密。

其次来看加密循环。给定一个类型 n 安全($n\in\{0,\cdots,7\}$)的加密方案 $\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$，可构造一个具有以下属性的类型 n 安全的加密方案 $\Pi'=(\mathcal{K},\mathcal{E}',\mathcal{D}')$：如果给敌手哪怕只有一个 $c\xleftarrow{R}\mathcal{E}'_k(k)$形式的密文，那么 Π' 也将变得完全不安全。不仅用 k 加密 k 是有问题的，而且更长的循环也会出问题。例如，一个加密方案是类型 3 安全的，那么在以下情况下它将变得不安全：用 a 加密 b，然后将 a 与 b 的角色倒过来，用 b 加密 a，因为将这两个密文并置起来，也许会很平凡地将 a 和 b 全都泄露。因此，这里只关注没有加密循环的表达式。

3. 加密方案安全性的定义(类型 0、1、3)

定义 4.14(类型 0 安全) 设 $\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$ 为一加密方案，$\eta\in$ Parameter 为一安全参数，A 为一敌手，定义

$$\begin{aligned}\mathrm{Adv}^{0}_{\Pi[\eta]}(A)\stackrel{\mathrm{def}}{=\!=}&\Pr[k,k'\xleftarrow{R}\mathcal{K}(\eta):A^{\mathcal{E}_k(\cdot),\mathcal{E}_{k'}(\cdot)}(\eta)=1]\\&-\Pr[k\xleftarrow{R}\mathcal{K}(\eta):A^{\mathcal{E}_k(0),\mathcal{E}_k(0)}(\eta)=1]\end{aligned}$$

如果对所有概率多项式时间的敌手 A，$\mathrm{Adv}^{0}_{\Pi[\eta]}(A)$是可忽略的，则称加密方案 Π 是类型 0 安全的。

在第一个概率中，括号中的量描述了执行一个实验后的事件。在该实验中，通过运行密钥生成算法$\mathcal{K}$，独立地产生两个密钥 k 和 k'，然后运行敌手 A，敌手具有两个应答器：一个左应答器 f 和一个右应答器 g。如果敌手向左应答器 f 请求一个查询 $m\in$ String，应答器返回一个用 k 对 m 的随机加密值，类似地，如果敌手向右应答器 g 请求一个查询 $m\in$ String，应答器返回一个用 k'对 m 的随机加密值。

在第二个概率中，其实验只用密钥生成算法$\mathcal{K}$生成了一个密钥。敌手也有两个应答器，一个左应答器 f 和一个右应答器 g。当请求一个查询 m 时，应答器忽略该查询，而是选取 $c\xleftarrow{R}\mathcal{E}_k(0)$，并返回 c。一个用 k 对 m 的随机加密值，类似地，如果敌手向右应答器 g 请求一个查询 $m\in$ String，应答器返回一个用 k'对 m 的随机加密值。

类型 0 优势是指以上两个概率的差。直观上，如果敌手不能根据其输入/输出行为将以上两种加密盒区别出来，那么相应的加密方案就是类型 0 安全的。

类似地，可以定义类型 1 安全和类型 3 安全。同定义 4.14，$\mathcal{E}_k(\cdot)$是一个应答器，对输入 m 返回 $c\xleftarrow{R}\mathcal{E}_k(m)$。$\mathcal{E}_k(0^{|\cdot|})$也是一个应答器，它对输入 m 返回 $c\xleftarrow{R}\mathcal{E}_k(0^{|m|})$。

定义 4.15(类型 1 安全) 设 $\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$ 为一加密方案，$\eta\in$ Parameter 为一安全参

数，A 为一敌手，定义

$$\mathrm{Adv}^{1}_{\Pi[\eta]}(A) \stackrel{\text{def}}{=} \Pr[k, k' \stackrel{R}{\leftarrow} \mathcal{K}(\eta) : A^{\mathcal{E}_k(\cdot), \mathcal{E}_{k'}(\cdot)}(\eta) = 1] - \Pr[k \stackrel{R}{\leftarrow} \mathcal{K}(\eta) : A^{\mathcal{E}_k(0^{|\cdot|}), \mathcal{E}_k(0^{|\cdot|})}(\eta) = 1]$$

如果对所有概率多项式时间的敌手 A，$\mathrm{Adv}^{1}_{\Pi[\eta]}(A)$是可忽略的，则称加密方案 Π 是类型 1 安全的。

定义 4.16（类型 1 安全） 设 $\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$为一加密方案，$\eta\in$ Parameter 为一安全参数，A 为一敌手，定义

$$\mathrm{Adv}^{3}_{\Pi[\eta]}(A) \stackrel{\text{def}}{=} \Pr[k \stackrel{R}{\longleftarrow} \mathcal{K}(\eta) : A^{\mathcal{E}_k(\cdot)}(\eta) = 1] - \Pr[k \stackrel{R}{\longleftarrow} \mathcal{K}(\eta) : A^{\mathcal{E}_k(0^{|\cdot|})}(\eta) = 1]$$

如果对所有概率多项式时间的敌手 A，$\mathrm{Adv}^{3}_{\Pi[\eta]}(A)$是可忽略的，则称加密方案 Π 是类型 3 安全的。

类型 3 安全是标准的，而类型 0 和类型 1 安全不是，但类型 0 和类型 1 安全可以通过类型 3 安全来构造。

4.5.3 形式等价的计算可靠性

本小节通过以下两步将两种密码观点关联起来：首先，表明给定一个加密方案 Π，如何将总体关联到一个表达式 M；然后表明在合适的假设下，表达式的等价可导致总体的不可区分性。

1. 将总体关联到表达式

令 $\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$为一个加密方案，$\eta\in$ Parameter 为一个安全参数，给每个形式化表达式 $M\in$ Exp 关联一个串$[\![M]\!]_{\Pi[\eta]}$上的分布，从而形成一个总体$[\![M]\!]_{\Pi}$。这种关联构成了表达式的一个具体语义。具体关联如下。

(1) 首先，将 M 中出现的每一个密钥符号 K 通过使用密钥生成器$\mathcal{K}(\eta)$映射到一个位串 $\tau(K)$。

(2) 将形式化位 0 和 1 映射到其标准串描述。

(3) 通过并置 M 和 N 的像来得到(M,N)的像。

(4) 通过计算$\mathcal{E}_{\tau(K)}(x)$得到形式化加密$\{M\}_K$ 的像，其中 x 为 M 的像。

(5) 在所有情况下，对串描述都加上其类型(即“key”，“bool”，“pair”，“ciphertext”)，以避免混淆。

更精确的定义如下：

```
algorithm INITIALIZE_η(M)
  for K ∈ Keys(M) do τ(K) ←R− 𝒦(η)
algorithm CONVERT(M)
  if M = K where K ∈ Keys then
    return ⟨τ(K),"key"⟩
  if M = b where b ∈ Bool then
    return ⟨b,"bool"⟩
```

if $M = (M_1, M_2)$ then
  return $\langle$CONVERT(M_1), CONVERT(M_2),"pair"$\rangle$
if $M = \{M_1\}_K$ then
  $x \xleftarrow{R}$ CONVERT(M_1)
  $y \xleftarrow{R} \mathcal{E}_{\tau(K)}(x)$
  return$\langle y$,"ciphertext"$\rangle$

其中 Keys(M)表示所有在 M 中出现的密钥符号,用$\langle x_1,\cdots,x_k\rangle$表示 $x_1,\cdots,x_k$ 的一般编码串。辅助的初始化过程将每个 Keys(M)中的密钥映射到一个的密钥 $\tau(K)$,一个串在$[\![M]\!]_{\Pi(\eta)}$中的概率可由算法 CONVERT(M)得出。

2. 等价性蕴涵不可区分性

下面来证明等价的表达式对应于总体的不可区分性(即 $M\cong N$ 蕴涵$[\![M]\!]_\Pi\approx[\![N]\!]_\Pi$)。

定理 4.5 设 M、N 为非循环表达式,Π 为类型 0 安全的加密方案,如果 $M\cong N$,则$[\![M]\!]_\Pi\approx[\![N]\!]_\Pi$。

证明 用混合论证的方法来证明。由于证明过程较复杂,在证明的同时用一个例子来说明。整个证明分为以下几个阶段。

(1) 密钥重命名。大致说来,该阶段的目标是通过密钥重命名的方式修改表达式 M 和 N,使得 pattern(M)=pattern(N)。另外,M 有隐藏密钥 $K_1,\cdots,K_m$,N 有隐藏密钥 $K_1,\cdots,K_n$,其中,仅当 $I\geqslant i$ 时有 K_I 加密 K_i。M 和 N 均有可获取(Recoverable)密钥 $J_1,\cdots,J_\mu$。

首先对 Keys(M)进行划分:

$$\text{recoverable}(M) = \{K \in \text{Keys}(M) \mid M\vdash K\}$$
$$\text{hidden}(M) = \text{Keys}(M) - \text{recoverable}(M)$$

设 $\mu=|\text{recoverable}(M)|$,$m=|\text{hidden}(M)|$,构造一个图 $G_M=(V_M,E_M)$,其顶点 $V_M=$ hidden(M),E_M 中有边 $K\rightarrow K'$当且令当 K 在 M 中加密 K'。M 的非循环性意味着 G_M 是非循环的,从而可以重命名 Keys(M)中的密钥,使得隐藏密钥为 $K_1,\cdots,K_m$,可获取密钥为 $J_1,\cdots,J_\mu$,且 $K_I\rightarrow K_i\in E_M$ 蕴涵着 $I\geqslant i$,也就是说,给 M 中层次越深的密钥关联一个越小的数字。设表达式 M'为通过以上重命名后所得到的结果。

例如,设 M 为表达式(这里省去了逗号和括号),用 $abcd$ 表示$(((a,b),c),d)$:

$$\{0\}_{K_6}\{K_1\ 1\}_{K_4}K_2\{0\}_{K_3}\{K_6\}_{K_4}\{K_1K_3\}_{K_4}\{1\ 1\ 1\}_{K_5}0\{K_1\}_{K_6}\{K_5\}_{K_2}$$

从而有:

$$\text{Keys}(M)=\{K_1,K_2,K_3,K_4,K_5,K_6\}$$
$$\text{recoverable}(M)=\{K_2,K_5\}$$
$$\text{hidden}(M)=\{K_1,K_3,K_4,K_6\}$$

图 $G_M=(V_M,E_M)$有顶点 $V_M=\{K_1,K_3,K_4,K_6\}$和边 $K_4\rightarrow K_1$,$K_4\rightarrow K_3$,$K_4\rightarrow K_6$,$K_6\rightarrow K_1$,重命名为$(K_1,K_2,K_3,K_4,K_5,K_6)$为$(K_1,J_1,K_2,K_4,J_2,K_3)$,得到一个新的表达式 M':$\{0\}_{K_3}\{K_1\ 1\}_{K_4}J_1\{0\}_{K_2}\{K_3\}_{K_4}\{K_1K_2\}_{K_4}\{1\ 1\ 1\}_{J_2}0\{K_1\}_{K_3}\{J_2\}_{J_1}$。

由于 $M\cong N$,根据重命名下等价的定义,存在 Keys(N)上的函数 σ 使得 pattern(M)=pattern($N\sigma$),在这两个模式中出现的密钥分别为 recoverable(M)和 recoverable($N\sigma$),从而它们是相等的。而且,由于 hidden(N)中的密钥被关在符号□中,所以 σ 在 hidden(N)上的取值是

无关紧要的。这样，根据非循环性，再将这些密钥重命名为 $K_1,\cdots,K_n$，使得仅当 $I\geqslant i$ 时有 K_I 加密 K_i。可以得到一个函数 σ' 和 N'，使得 $N'=N\sigma'$，$M'\equiv N'$，

$$\text{recoverable}(M')=\text{recoverable}(N')=\{J_1,\cdots,J_\mu\},$$

$$\text{hidden}(N')=\{K_1,\cdots,K_n\}，\text{当 } I\geqslant i \text{ 时}，K_I \text{ 在 } N' \text{ 中加密 } K_i。$$

继续前面的例子，令 N 为表达式

$$\{1\ 1\}_{K_2}\{K_3\}_{K_2}K_1\{K_3\}_{K_2}\{K_8\}_{K_2}\{1\}_{K_5}\{1\ 1\ 1\}_{K_3}0\{0\ 0\}_{K_8}\{K_3\}_{K_1}$$

从而，$\text{recoverable}(N)=\{K_1,K_3\}$，$\text{hidden}(N)=\{K_2,K_5,K_8\}$，重命名 (K_1,K_2,K_3,K_5,K_8) 为 (J_1,K_3,J_2,K_1,K_2)，得到相应的表达式 N'：

$$\{1\ 1\}_{K_3}\{J_2\}_{K_3}J_1\{J_2\}_{K_3}\{K_2\}_{K_3}\{1\}_{K_1}\{1\ 1\ 1\}_{J_2}0\{0\ 0\}_{K_2}\{J_2\}_{J_1}$$

注意：$N'_{(N)}$ 和 $M'_{(M)}$ 中的隐藏密钥数不同，但可获取密钥数是相同的。

(2) 模式混合。在本阶段，引入模式 $M_0,M_1,\cdots,M_m$ 和 $N_0,N_1,\cdots,N_n$，使得这些模式在 M' 和 N' 之间形成一个链，使用前面定义的 p 函数，令

$$M_i=p(M',\text{recoverable}(M')\cup\{K_1,\cdots,K_i\})$$

其中，$K_1,\cdots,K_m$ 为 M' 中的隐藏密钥，$i\in\{0,\cdots,m\}$。特别地，有 $M_0=\text{pattern}(M')$，$M_m=M'$。类似地，对于 $j\in\{0,\cdots,n\}$，令

$$N_j=p(N',\text{recoverable}(N')\cup\{K_1,\cdots,K_j\})$$

特别地，$N_0=\text{pattern}(N')$，$N_n=N'$。直观地说，M_i 和 N_i 为假设攻击者拥有关于隐藏密钥的先验知识 $K_1,\cdots,K_i$ 时，分别在在 M' 和 N' 中所看到的模式，密钥的顺序保证了利用这些知识不能发现其他的隐藏密钥。

在前面的例子中，有

M'

‖

M_4:	$\{0\}_{K_3}$	$\{K_1\ 1\}_{K_4}$	J_1	$\{0\}_{K_2}$	$\{K_3\}_{K_4}$	$\{K_1,K_2\}_{K_4}$	$\{1\ 1\ 1\}_{J_2}\ 0$	$\{K_1\}_{K_3}$	$\{J_2\}_{J_1}$
M_3:	$\{0\}_{K_3}$	□	J_1	$\{0\}_{K_2}$	□	□	$\{1\ 1\ 1\}_{J_2}\ 0$	$\{K_1\}_{K_3}$	$\{J_2\}_{J_1}$
M_2:	□	□	J_1	$\{0\}_{K_2}$	□	□	$\{1\ 1\ 1\}_{J_2}\ 0$	□	$\{J_2\}_{J_1}$
M_1:	□	□	J_1	□	□	□	$\{1\ 1\ 1\}_{J_2}\ 0$	□	$\{J_2\}_{J_1}$
M_0:	□	□	J_1	□	□	□	$\{1\ 1\ 1\}_{J_2}\ 0$	□	$\{J_2\}_{J_1}$
‖									
N_0:	□	□	J_1	□	□	□	$\{1\ 1\ 1\}_{J_2}\ 0$	□	$\{J_2\}_{J_1}$
N_1:	□	□	J_1	□	□	$\{1\}_{K_1}$	$\{1\ 1\ 1\}_{J_2}\ 0$	□	$\{J_2\}_{J_1}$
N_2:	□	□	J_1	□	□	$\{1\}_{K_1}$	$\{1\ 1\ 1\}_{J_2}\ 0$	$\{0\ 0\}_{K_2}$	$\{J_2\}_{J_1}$
N_3:	$\{11\}_{K_3}$	$\{J_2\}_{K_3}$	J_1	$\{J_2\}_{K_3}$	$\{K_2\}_{K_3}$	$\{1\}_{K_1}$	$\{1\ 1\ 1\}_{J_2}\ 0$	$\{0\ 0\}_{K_2}$	$\{J_2\}_{J_1}$

‖

N'

注意，$\text{pattern}(M')=M_0=N_0=\text{pattern}(N')$。

(3) 定义模式的总体。接下来，将每一个模式 $M_0,\cdots,M_m,N_0,\cdots,N_n$ 分别映射到一个总体 $[M_0]_\Pi,\cdots,[M_m]_\Pi,[N_0]_\Pi,\cdots,[N_n]_\Pi$ 上，该映射可通过扩展前文的 CONVERT 算法而得到，使其不仅可用于表达式，而且可用于模式。这个扩展很简单：任何时间遇到符号□，它用一个新的、固定的密钥（除此之外，该密钥没有其他用处）对 0 进行加密，并将加密结果返

回。具体来说，在 Initialize 中加入下面一行：

$$\tau(K_0) \xleftarrow{R} \mathcal{K}(\eta)$$

在 CONVERT 中加入以下几行：

$$\text{if } M = \square \text{ then}$$

$$y \xleftarrow{R} \mathcal{E}_{\tau(K_0)}(0)$$

$$\text{return}\langle y, \text{“ciphertext”}\rangle$$

（4）寻找大间隙。由于 M 和 M' 的不同仅在于其中下标的不同，显然有 $[\![M]\!]_\Pi = [\![M']\!]_\Pi$，类似地，有 $[\![N]\!]_\Pi = [\![N']\!]_\Pi$，因此，目标就是要证明 $[\![M']\!]_\Pi \approx [\![N']\!]_\Pi$。

用反证法：为了得到与 Π 为类型 0 安全的相矛盾的结果，假设存在一个敌手 A 能够区分 $[\![M']\!]_\Pi$ 和 $[\![N']\!]_\Pi$。根据定义，A 运行多项式时间，且

$$\lambda(\eta) = \Pr[y \xleftarrow{R} [\![M']\!]_\Pi : A(\eta, y) = 1] - \Pr[y \xleftarrow{R} [\![N']\!]_\Pi : A(\eta, y) = 1]$$

是不可忽略的，也就是说，存在常数 c，存在无穷集 $\mathcal{N}$，对所有 $\eta \in \mathcal{N}$ 有 $\lambda(\eta) > \eta^{-c}$。设 $0 \leqslant i \leqslant m$，$1 \leqslant j \leqslant n$，定义

$$p_i(\eta) = \Pr[y \xleftarrow{R} [\![M_i]\!]_{\Pi[\eta]} : A(\eta, y) = 1]$$

$$q_j(\eta) = \Pr[y \xleftarrow{R} [\![N_j]\!]_{\Pi[\eta]} : A(\eta, y) = 1]$$

简洁起见，下文有时会省略参数 η。由于 $M' = M_m$，$N' = N_n$，有 $\lambda = p_m - q_n$。另外，由于 M' 和 N' 所生成的模式相同，所以还有 $p_0 = q_0$。由此，可得到下式：

$$\begin{aligned}\lambda = &(p_m - p_{m-1}) + (p_{m-1} - p_{m-2}) + \cdots + (p_1 - p_0) \\ &+ (q_0 - q_1) + (q_1 - q_2) + \cdots + (q_{n-1} - q_n)\end{aligned}$$

这样，就将 λ 表示成 $m+n$ 项的和。根据三角不等式，或者存在 $i \in \{1, \cdots, m\}$ 使得 $p_i - p_{i-1} \geqslant \lambda/(m+n)$，或者存在 $j \in \{1, \cdots, n\}$，使得 $q_{j-1} - q_j \geqslant \lambda/(m+n)$。而且，对每一个 $\eta \in \mathcal{N}$，都存在这样一个合适的下标 i 或 j，由于和式项的数目是有限的、固定的，从而存在一个对无穷多的 $\eta \in \mathcal{N}$ 都适用的这种下标 i 或 j，由于下标为 i 或为 j 的情况是类似的，不妨取该下标为 i。因此，存在一个无穷集 $\mathcal{N}' \subseteq \mathcal{N}$，对每一个 $\eta \in \mathcal{N}'$，有 $p_i(\eta) - p_{i-1}(\eta) \geqslant \lambda(\eta)/(m+n)$。

在例子中，假设存在敌手可以以 0.5 的优势区分 $[\![M_4]\!]_{\Pi[\eta]}$ 和 $[\![N_3]\!]_{\Pi[\eta]}$，即 $[\![M]\!]_{\Pi[\eta]}$ 和 $[\![N]\!]_{\Pi[\eta]}$，从而敌手至少会以 0.50/7 的优势区分以下各对之一：$[\![M_4]\!]_{\Pi[\eta]}$ 和 $[\![M_3]\!]_{\Pi[\eta]}$；$[\![M_3]\!]_{\Pi[\eta]}$ 和 $[\![M_2]\!]_{\Pi[\eta]}$；$[\![M_2]\!]_{\Pi[\eta]}$ 和 $[\![M_1]\!]_{\Pi[\eta]}$；$[\![M_1]\!]_{\Pi[\eta]}$ 和 $[\![M_0]\!]_{\Pi[\eta]}$；$[\![N_0]\!]_{\Pi[\eta]}$ 和 $[\![N_1]\!]_{\Pi[\eta]}$；$[\![N_1]\!]_{\Pi[\eta]}$ 和 $[\![N_2]\!]_{\Pi[\eta]}$；$[\![N_2]\!]_{\Pi[\eta]}$ 和 $[\![N_3]\!]_{\Pi[\eta]}$。例如，假设该对为 $[\![M_3]\!]_{\Pi[\eta]}$ 和 $[\![M_2]\!]_{\Pi[\eta]}$，则当从 $[\![M_3]\!]_{\Pi[\eta]}$ 中抽样时 A 回答 1 的频率将实质性地大于从 $[\![M_2]\!]_{\Pi[\eta]}$ 中抽样时 A 回答 1 的频率。

（5）与 Π 的类型 0 安全相矛盾。假设 A 可区分 $[\![M_i']\!]_\Pi$ 和 $[\![M_{i-1}']\!]_\Pi$，通过使用 A，可构造一个敌手 A_0，来破坏 Π 的类型 0 安全性。A_0 可由以下算法定义：

算法 $A_0^{f,g}(\eta)$

$$\text{for } K \in \text{Keys}(M') \text{ do } \tau(K) \xleftarrow{R} \mathcal{K}(\eta)$$

$$y \xleftarrow{R} \text{CONVERT2}(M')$$

$b \xleftarrow{R} A(\eta, y)$
return b

算法 CONVERT2(M^*)
if $M^* = K$ where $K \in$ Keys then
return⟨$\tau(K)$,“Key”⟩
if $M^* = b$ where $b \in$ Bool then
return⟨b,“bool”⟩
if $M^* = (M_1^*, M_2^*)$ then
return⟨CONVERT2(M_1^*),CONVERT2(M_2^*),“pair”⟩
if $M^* = \{M_1^*\}_K$ then
if $K \in \{J_1, \cdots, J_\mu, K_1, \cdots, K_{i-1}\}$ then
$x \xleftarrow{R}$ CONVERT2(M_1^*)
$y \xleftarrow{R} \mathcal{E}_{T(K)}(x)$
return⟨y,“ciphertext”⟩
else if $K = K_i$ then
$x \xleftarrow{R}$ CONVERT2(M_1^*)
$y \xleftarrow{R} f(x)$
return ⟨y, “ciphertext”⟩
else if $K \in \{K_{i+1}, \cdots, K_m\}$ then
$y \xleftarrow{R} g(0)$
return ⟨y, “ciphertext”⟩

在该算法中,A_0 访问了两个应答器,f 和 g,它们可分别进行两种实例化:第一种情形是将$\mathcal{E}_{k_i}(\cdot)$作为应答器 f,其中随机选择 $k_i \xleftarrow{R} \mathcal{K}(\eta)$,将$\mathcal{E}_{k_0}(\cdot)$作为应答器 g,其中随机选择 $k_0 \xleftarrow{R} \mathcal{K}(\eta)$;另一种情形下$\mathcal{E}_{k_0}(0)$既作为 f 又作为 g,其中随机选择 $k_0 \xleftarrow{R} \mathcal{K}(\eta)$。这时,有

$$p_i(\eta) = \Pr[k_i, k_0 \xleftarrow{R} \mathcal{K}(\eta) : A_0^{\mathcal{E}_{k_i}(\cdot), \mathcal{E}_{k_0}(\cdot)}(\eta) = 1]$$

$$p_{i-1}(\eta) = \Pr[k_0 \xleftarrow{R} \mathcal{K}(\eta) : A_0^{\mathcal{E}_{k_0}(0), \mathcal{E}_{k_0}(0)}(\eta) = 1]$$

因为当 $f = \mathcal{E}_{k_i}(\cdot)$,$g = \mathcal{E}_{k_0}(\cdot)$时,CONVERT2($M'$)返回一个$[\![M_i]\!]_\Pi$的抽样,而当$f = \mathcal{E}_{k_0}(0)$,$g = \mathcal{E}_{k_0}(0)$时,CONVERT2($M'$)返回一个$[\![M_{i-1}]\!]_\Pi$的抽样,所以以上等式是成立的。注意,在两种情况下,用可获取密钥 K 的加密均对应于用相应 $\tau(K)$的加密,用在$\{K_1, \cdots, K_{i-1}\}$中的隐藏密钥 K 的加密同样对应于用 $\tau(K)$的加密,而用在$\{K_{i+1}, \cdots, K_n\}$中的隐藏密钥的加密将导致用 k_0 对 0 加密。对第一个等式来说,用的隐藏密钥 K_i 的加密同样对应于用 k_i 的加密,对第二个等式来说,用隐藏密钥 K_i 的加密将导致用 k_0 对 0 加密。由此,还有

$$\begin{aligned}\mathrm{Adv}^0_{\Pi[\eta]}(A_0) = &\Pr[k_i, k_0 \xleftarrow{R} \mathcal{K}(\eta) : A_0^{\mathcal{E}_{k_i}(\cdot), \mathcal{E}_{k_0}(\cdot)}(\eta) = 1] \\ &- \Pr[k_0 \xleftarrow{R} \mathcal{K}(\eta) : A^{\mathcal{E}_{k_0}(0), \mathcal{E}_{k_0}(0)}(\eta) = 1]\end{aligned}$$

$$= p_i(\eta) - p_{i-1}(\eta)$$

对无限多的 η(在$\mathcal{N}'$中大于$(m+n)$),可以得到

$$\begin{aligned} \mathrm{Adv}^0_{\Pi[\eta]}(A_0) &\geqslant \lambda(\eta)/(m+n) \\ &> \eta^{-c}/(m+n) \\ &> \eta^{-(c+1)} \end{aligned}$$

从而,$\mathrm{Adv}^0_{\Pi[\eta]}(A_0)$是不可忽略的。不出所料,该结论与假设 Π 是类型 0 安全的相矛盾。

继续完成前面的例子,假如从$[\![M_3]\!]_{\Pi[\eta]}$中抽样时 A 回答 1 的频率将实质性地大于从$[\![M_2]\!]_{\Pi[\eta]}$中抽样时 A 回答 1 的频率,可通过构造一个成功的敌手 A_0 来表明 Π 不是类型 0 安全的。该敌手分别依赖于两种被实例化的应答器 f 和 g,这两种实例来自于类型 0 安全的定义。第一种实例中,A_0 从$[\![M_3]\!]_{\Pi[\eta]}$中抽样,然后调用 A,第二种实例中,A_0 从$[\![M_2]\!]_{\Pi[\eta]}$中抽样,然后调用 A。因此,A_0 回答 1 的次数将实质性地大于第一种情况。

定理 4.5 给出了一种渐近的安全声明。其实,从其证明过程来看,不难找到一个相应的具体安全声明。

4.5.4 不完备性

也许有人会问,定理 4.5 的逆命题成立吗?也就是说,不可区分性蕴涵等价性吗?答案是否定的。原因很简单,如果应用算法时,所给的表达式 M 和 M'引起加密方案 Π 对其明文空间 Plaintext 之外的串进行加密,那么,即使 M 和 M'不等价,给它们所关联的总体也将是相同的。

这说明,在输入反常的情况下定理 4.5 的逆是不成立的,那么是不是将加密的输入限定在明文空间之内,以上逆命题就成立了呢?答案也是否定的。本节将提供一个自然的反例来避免以上提到的反常情形,但同样可得到否定的结论,即 AR 逻辑是不完备的。

为方便描述,用 BLOCK 来代替 AR 逻辑中的 BOOL 集,它表示分组符号的有限集合,每个符号表示一个固定长度的位串序列(BOOL 中仅包含了表示 0 和 1 的符号),相应地,模式的定义中涉及 BOOL 的量也用 BLOCK 来代替。另外,将函数 recoverable 建立在另一函数 F_{kr}上,F_{kr}定义在输入$(E,T)\in \mathrm{Exp}\times\mathcal{P}(\mathrm{Keys})$上,它返回仅用集合 T 中的密钥可从 E 中恢复出来的密钥集。这样,可以将 recoverable(E)定义为从表达式 E 中可恢复出的密钥,如果考虑集合包含序关系,可将 recoverable(E)看做函数 $F_{\mathrm{kr}}(E,\cdot)$的最小不动点。从算法的角度,该集合可很容易地以以下方式得到:给一个表达式 E,归纳定义集合 $G_0(E)$,$G_1(E)$,$G_2(E)$,…,其中 $G_i(E)$为(从不用密钥开始)i 次应用 $F_{\mathrm{kr}}(E,\cdot)$恢复出的密钥,然后取 recoverable$(E)=\bigcup_i G_i(E)$。注意,所有的 $G_i(E)$都是 E 中所出现的密钥符号集的子集,且有 $\phi=G_0(E)\subseteq G_1(E)\subseteq\cdots\subseteq G_i(E)\subseteq\cdots$。以上提到的几种函数可递归定义如下:

(1) $F_{\mathrm{kr}}(B,T)=T$;

(2) $F_{\mathrm{kr}}(K,T)=\{K\}\cup T$;

(3) $F_{\mathrm{kr}}((E_0,E_1),T)=F_{\mathrm{kr}}(E_0,T)\cup F_{\mathrm{kr}}(E_1,T)$;

(4) $F_{\mathrm{kr}}(\{E\}_K,T)=T$, if $K\notin T$;

(5) $F_{\mathrm{kr}}(\{E\}_K,T)=F_{\mathrm{kr}}(E,T)$, if $K\in T$;

① $G_0(E)=\phi$

② $G_i(E)=F_{\mathrm{kr}}(E,G_{i-1}(E))$

③ recoverable$(E)=\bigcup_i G_i(E)=G_{|E|}(E)$

以下给出不完备的反例。考虑两种位串的表示，一种显式地附加了类型标签，另一种没有附加类型标签，在两种情况下加密函数的输入都在所描述的范围之内。

(1) 无标签描述。设$\{F_i\}_{i\in I_\eta}$为伪随机函数簇，其中$F_i:\{0,1\}^\eta\to\{0,1\}^\eta$，$I_\eta=\{0,1\}^\eta$，定义明文空间$P=\{0,1\}^\eta$上的加密方案$\Pi'=(\mathcal{K}',\mathcal{E}',\mathcal{D}')$如下。

① 密钥生成算法$\mathcal{K}'$：输入η，$\mathcal{K}'$随机地从I_η中选择一个元素(即密钥生成算法是从簇中选择的一个随机函数)。

② 加密函数$\mathcal{E}'$：用$k\in I_\eta$对$m\in P$进行加密得到$\mathcal{E}'_k(m)=r\parallel(F_k(r)\oplus m)$，其中$r$是新生成的随机数，$\parallel$表示串的并置，$\oplus$表示位的异或操作。为简化起见，以下将省略括号，并假设$\oplus$的优先级高于$\parallel$。

③ 解密函数$\mathcal{D}'$：对于密文$(r\parallel C)$，其解密结果为$\mathcal{D}'_k(r\parallel C)=F_k(r)\oplus C$。

可以验证，对于密钥k和消息m，有$\mathcal{D}'_k(\mathcal{E}'_k(m))=m$，而且，在函数簇$\{F_i\}_{i\in I_\eta}$是伪随机的假设下，该方案是类型 0 安全的。

现在，考虑两个表达式：$E_1=(\{K_3\}_{K_1},K_1)$及$E_2=(\{K_3\}_{K_1},K_2)$，其模式如下。

$$\text{pattern}(E_1)=(\{K_3\}_{K_1},K_1)$$

$$\text{pattern}(E_2)=(\Box,K_2)$$

这两个表达式即使是在重命名的情况下也不等价，也就是说，在所给逻辑中不等价。现在，考虑在安全参数为η时这两个表达式所关联的概率分布$(r_1\parallel F_{k_1}(r_1)\oplus k_3,k_1)$及$(r_2\parallel F_{k_1}(r_2)\oplus k_3,k_2)$，其中，$r_1$、$r_2$、$k_1$、$k_2$、$k_3$是$\{0,1\}^\eta$上相互独立的均匀分布，这两个分布是相同的。第一个表达式的分布中，开头部分和结尾部分分别为r_1和k_1，它们是随机的，而且是独立的，中间部分$F_{k_1}(r_1)\oplus k_3$将固定值$F_{k_1}(r_1)$与一个独立的随机值k_3相异或，从而它也是随机的。类似地，第二个表达式的分布也是随机的。可见，虽然在逻辑中这两个表达式不等价，但其分布却是相同的。

形式化语义和计算语义的差别可以非形式化地表述如下：在形式化语义中，隐式地假设了只有当解密密钥为k时，才可以成功地解密$\{M\}_k$，而且对于解密来说只有两种结果，要么成功，要么失败。通常在加密方案中没有这种假设，因为拥有一个密钥k并不能帮助一个人区分出一个密文是用k加密的，还是用另一个与k不同的k'加密的。注意，如果使用了类型标签，这一反例就不成其为反例了。这时，密钥符号k的像成为$\langle\mathcal{T}(k)$,“key”$\rangle$，对这类消息的密文进行解密，通过检查标签便可查出是否使用了错误的密钥进行解密。下一个反例将表明，使用类型标签还是不能解决问题，至少对于有限明文空间的加密方案来说是这样的。

(2) 标签描述。用以上定义的加密方案$\Pi'=(\mathcal{K}',\mathcal{E}',\mathcal{D}')$构造一个新的加密方案$\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$，其明文空间为$P=\{\langle m$,“key”$\rangle\mid m\in\{0,1\}^\eta\}$，其中$\eta$为安全参数。密钥生成算法$\mathcal{K}$与$\mathcal{K}'$相同；加密算法$\mathcal{E}_k(\langle m$,“key”$\rangle)=\mathcal{E}'_k(m)$，即用$k$对$\langle m$,“key”$\rangle\in P$加密时，先忽略标签，然后直接用$\mathcal{E}'_k(\cdot)$加密即可；对密文$c$解密时，$\mathcal{D}_k(c)=\langle\mathcal{D}'(c)$,“key”$\rangle$，即先直接用$\mathcal{D}'_k(\cdot)$解密，然后在结果后面加上标签“key”后再输出。不难检验，当只对P中的消息进行加密时，这种方案是正确的，而且是类型 0 安全的。这时再来看两个表达式所对应的分布：$(\langle r_1\parallel F_{k_1}(r_1)\oplus k_3$,“ciphertext”$\rangle$, $\langle k_1$,“key”$\rangle)$和$(\langle r_2\parallel F_{k_1}(r_2)\oplus k_3$,“ciphertext”$\rangle)$, $(\langle k_2$,“key”$\rangle)$，其中r_1、r_2、k_1、k_2、k_3是$\{0,1\}^\eta$上相互独立的均匀分布。可以看出这两个

分布还是相同的。

4.5.5 完备性定理

上一小节表明了类型 0 的安全性不足以支持 AR 逻辑的完备性。所给出的反例还表明，不支持的原因在于对于给定的密文和密钥，不能确定该密文是否是用该密钥加密的。本小节对这一想法作进一步探讨：当给加密方案附加一定的条件时，AR 逻辑就是完备的了。将该条件称为无混淆，并表明标准的加密方案就是无混淆的。

1. *无混淆加密与认证加密*

无混淆属性可非形式化地描述如下：随机地、独立地生成两个密钥，当用其中一个密钥加密时，用另一个密钥将只能以可忽略的概率解密。

定义 4.17(无混淆) 设 $D=\{D_1(\eta),D_2(\eta),\cdots,D_l(\eta)\}$ 为分布总体的有限集，Π 是安全参数为 η 的对称加密方案，称 Π 对 D 来说是无混淆的，当且仅当存在一个可忽略函数 v_D 使得对任意 $1\leqslant i\leqslant l$ 有

$$\Pr[k_1,k_2 \stackrel{R}{\leftarrow} \mathcal{K}(\eta),x \stackrel{R}{\leftarrow} D_i(\eta):\mathcal{D}_{k_1}(\mathcal{E}_{k_2}(x))\neq\perp]\leqslant v_D(\eta) \tag{4-13}$$

如果 Π 对任意分布总体来说都是无混淆的，则称 Π 是无混淆的。

认证加密的概念可形式化地表达如下：多项式时间的敌手甚至在看到过多项式多个随机选择的消息的密文后，也只能以可忽略的概率生成有效的密文(未解密的密文用$\perp$表示)。此处主要介绍明文的完整性，或称之为 INT-PTXT 安全，定义如下。

定义 4.18(安全认证加密方案) 设 $\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$为一对称加密方案，针对 Π 的认证性的敌手为一多项式时间算法 A，它可以访问加密应答器$\mathcal{E}_k(\cdot)$，应答器对每次敌手的查询 m，以独立的随机加密方式返回$\mathcal{E}_k(m)$。认证加密方案的安全性可用以下实验来定义：随机生成密钥 k(使用安全参数 η)，并用$\mathcal{E}_k(\cdot)$实例化地加密应答器。然后让敌手与应答器交互，交互结束时由敌手输出一个伪造的密文 c(要求敌手从未向应答器询问过$\mathcal{D}_k(c)$的密文)，如果有$\mathcal{D}_k(c)\neq\perp$(即 c 为有效密文)，就认为敌手成功。如果敌手成功的概率为 η 的一个可忽略函数，则称该方案具有认证性。

以下引理建立了安全认证加密方案与无混淆之间的关系。

引理 4.4 设 $\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$为一个安全认证加密方案，那么对于任意多项式时间可抽样分布的有限集来说，Π 是无混淆的。

证明 设 Π 为一个安全认证加密方案，$D=\{D_1(\eta),D_2(\eta),\cdots,D_l(\eta)\}$为多项式时间可抽样分布的有限集，对每一个 $1\leqslant i\leqslant l$，定义一个针对 Π 的认证性的敌手 A_i，并给 A_i 提供一个应答器$\mathcal{E}_k$，其中，k 是随机选择的，且 A_i 不知道 k。但在此处 A_i 并不引用应答器，它只是以新鲜的随机生成因子通过运行$\mathcal{K}$生成另一个随机密钥 k_i，按 D_i 分布选择一个随机的明文 x，并输出 $c=\mathcal{E}_{k_i}(x)$作为伪造密文。由于 A_i 没有询问过应答器，根据 Π 的安全性，其成功的概率 $v_i(\eta)=\Pr[k \stackrel{R}{\leftarrow}\mathcal{K}(\eta),c \stackrel{R}{\leftarrow}A_i(\eta):\mathcal{D}_k(c)\neq\perp)]$是安全参数 η 的可忽略函数。令 $v_D(\eta)=\max\limits_{1\leqslant i\leqslant l} v_i(\eta)$，不难看出，$v_D$ 也是一个可忽略函数。根据 v_D 和 A_i 的定义立即可知，对所有$i=1,\cdots,l$，分布总体 D_i 的混淆概率至多为 $\Pr[k \stackrel{R}{\leftarrow}\mathcal{K}(\eta),k_i \stackrel{R}{\leftarrow}\mathcal{K}_i(\eta),x \stackrel{R}{\leftarrow}D_i(\eta):\mathcal{D}_k(\mathcal{E}_{k_i}(x))\neq\perp)]\leqslant v_i(\eta)\leqslant v_D(\eta)$。

2. 完备性定理

定理 4.6　设 Π 为一个类型 0 的安全认证加密方案，或者更一般地，一个类型 0 安全的无混淆加密方案。如果 E_0 和 E_1 为非循环表达式，那么

$$E_0 \cong E_1 \text{ 当且仅当} [\![E_0]\!]_{\Pi(\eta)} \approx [\![E_1]\!]_{\Pi(\eta)}$$

"仅当"方向为可靠性结果，前文已证，"当"方向为完备性结果，将在此证明。

证明思路：主要要表明从 E 的位串描述(即抽样 $e \xleftarrow{R} [\![E]\!]_{\Pi(\eta)}$)中可以以很高的概率恢复出形式化描述下 E 的模式。为此，引入分别对应于函数 F_{kr} 和 recoverable，但定义在位串上，而不是表达式上的函数，然后证明利用新构造函数从 E 的位串中得到的模式，与从 E 本身在可换名情况下得到的模式是等价的。

恢复密钥：设 E 为固定的表达式，$e \in [\![E]\!]_{\Pi(\eta)}$ 为相应分布上的随机抽样，τ 为密钥赋值，则从位串中可恢复出的密钥集合为

$$\tau(\text{recoverable}(E)) = \{\tau(K) \mid K \in \text{recoverable}(E)\}$$

可用以下算法定义函数 C_{kr}: String $\times \mathcal{P}$(Key) $\to \mathcal{P}$(Key) 和 Crecoverable: String $\to \mathcal{P}$(Key)。直观地看，它们分别是函数 F_{kr} 和 recoverable 在计算模型下相对应的函数，其中的一些表示不难在形式化模型下找到其相对应的表示。

```
Algorithm C_kr(e,tT)
  if e=⟨b,"block"⟩output tT
  if e=⟨k,"key"⟩output tT∪{k}
  if e=⟨(m,n),"pair"⟩output C_kr(m,tT)∪C_kr(n,tT)
  if e=⟨c,"ciphertext"⟩then
    if for exactly one k∈tT,D_k(y)≠⊥
      then output, C_kr(D_k(c),tT)
      else output tT
```

```
Algorithm Crecoverable(e)
  T_0←ϕ;cont←true
  while cont
    T_{i+1}←C_kr(e,T_i)
    if T_{i+1}=T_i then cont←false
      else i←i+1
  output T_i
```

引理 4.5　设 Π 为一个无混淆的加密方案，$E \in$ Exp，$T \subseteq$ Keys。e 为根据分布$[\![E]\!]_{\Pi(\eta)}$选择的一个位串，τ 为密钥赋值，那么存在可忽略函数 v，使得

$$\Pr[C_{kr}(e,\tau(T)) \neq \{\tau(F_{kr}(E,T))\}] \leqslant |E\| T| v(\eta) \tag{4-14}$$

证明　设 v 为与表达式 E 相关联的可忽略函数，通过对 E 的结构使用归纳法来证明。

如果 $E=B$，且 $B \in$ Block，则 e 具有$\langle b$,"block"$\rangle$的形式，其中 b 为分组 B 的期望描述。这时有 $F_{kr}(E,T)=T|C_{kr}(e,\tau(T))=\tau(T)=\tau(F_{kr}(E,T))$，因此

$$\begin{aligned}\Pr[C_{kr}(e,\tau(T)) \neq \{\tau(F_{kr}(E,T))\}] &= 1-\Pr[C_{kr}(e,\tau(T)) \\ &= \tau(F_{kr}(E,T))] = 0\end{aligned}$$

如果 $E=K$，则 e 具有$\langle\tau(K)$,"key"$\rangle$的形式，根据 F_{kr} 和 C_{kr} 的定义，有

$$\begin{aligned}F_{kr}(E) = \{K\} \cup T \text{ and } C_{kr}(e,\tau(T)) &= \tau(T) \cup \tau\{K\} \\ &= \tau(T \cup \{K\}) = \tau(F_{kr}(E,T))\end{aligned}$$

从而

$$\Pr[C_{kr}(e,\tau(T)) \neq \tau(F_{kr}(E,T))] = 1-\Pr[C_{kr}(e,\tau(T)) = \tau(F_{kr}(E,T))] = 0$$

如果 $E=(E_0,E_1)$，那么 e 具有 $e=\langle(e_0,e_1)$,"pair"$\rangle$的形式，其中 $e_i \xleftarrow{R} [\![E_i]\!]_{\Pi(\eta)}$，$i=0,1$。

根据 F_{kr} 和 C_{kr} 的定义，有

$$F_{kr}(E)=F_{kr}(E_0,T)\cup F_{kr}(E_1,T) \text{ and } C_{kr}(e,\tau(T))$$
$$=C_{kr}(e_1,\tau(T))\cup C_{kr}(e_2,\tau(T))$$

由此，有以下不等式：

$$\Pr[C_{kr}(e_0,\tau(T))\neq\tau(F_{kr}(E,T))]$$
$$\leqslant\Pr[C_{kr}(e_0,\tau(T))\neq\tau(F_{kr}(E_0,T))\text{or } C_{kr}(e_1,\tau(T))\neq\tau(F_{kr}(E_1,T))]$$
$$\leqslant\Pr[C_{kr}(e_0,\tau(T))\neq\tau(F_{kr}(E_0,T))]+\Pr[C_{kr}(e_1,\tau(T))\neq\tau(F_{kr}(E_1,T)])$$

结合归纳假设，有

$$\Pr[C_{kr}(e,\tau(T))\neq\tau(F_{kr}(E,T))]\leqslant|E_0||T|v(\eta)+|E_1||T|v(\eta)$$
$$=(|E_0|+|E_1|)|T|v(\eta)=|E||T|v(\eta)$$

如果 $E=\{E_0\}_K$，那么 e 具有 $\langle\mathcal{E}_{\tau(K)}(e_0)$，"ciphertext"$\rangle$ 的形式，其中 $e_0\xleftarrow{R}[\![E_0]\!]_{\Pi(\eta)}$。分以下两种情况。

(1) 如果 $K\notin T$，根据 F_{kr} 的定义，有 $F_{kr}(E,T)=T$。根据 C_{kr} 的定义，为了使集合 $C_{kr}(e,\tau(T))$ 与 $\tau(T)$ 不同，则必须成功地用 $\tau(T)$ 中的一个密钥对 $\mathcal{E}_{\tau(K)}(e_0)$ 进行解密。然而这将违反 Π 的安全性。形式化地，

$$\Pr[C_{kr}(e,\tau(T))\neq\tau(F_{kr}(E,T))]\leqslant\Pr[\exists k\in\tau(T)\cdot\mathcal{D}_k(\mathcal{E}_{\tau(K)}(e_0))\neq\perp]$$
$$\leqslant\sum_{k\in\tau(T)}\Pr[\mathcal{D}_k(\mathcal{E}_{\tau(K)}(e_0))\neq\perp]$$

由于 $K\notin T$，$\tau(K)$ 与 $k\in\tau(T)$ 是独立生成的，根据 Π 的无混淆性，有 $\Pr[\mathcal{D}_k(\mathcal{E}_{\tau(k)}(e_0)\neq\perp]\leqslant v(\eta)$，从而可得

$$\Pr[C_{kr}(e,\tau(T))\neq\tau(F_{kr}(E,T))]\leqslant|T|v(\eta)\leqslant|E||T|v(\eta)$$

(2) 如果 $K\in T$，根据 F_{kr} 的定义，有 $F_{kr}(\{E\}_K,T)=F_{kr}(E,T)$。根据 C_{kr} 的定义，有以下不等式

$$\Pr[C_{kr}(e,\tau(T))=\tau(F_{kr}(E,T))]$$
$$\geqslant\Pr[C_{kr}(e_0,\tau(T))=\tau(F_{kr}(E_0,T))\text{ and }\forall k_i\in\tau(T)\backslash\{\tau(K)\}\cdot\mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0))=\perp]$$
$$=1-\Pr[C_{kr}(e_0,\tau(T))\neq\tau(F_{kr}(E_0,T))\text{ or }\exists k_i\in\tau(T)\backslash\{\tau(K)\}\cdot\mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0))\neq\perp]$$
$$\geqslant 1-\Big(\Pr[C_{kr}(e_0,\tau(T))\neq\tau(F_{kr}(E_0,T))]+\sum_{k_i\in\tau(T)\backslash\{\tau(K)\}}\Pr[\mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0))\neq\perp]\Big)$$

由于 Π 是安全认证方案，因而，对任意 $k_i\in\tau(T)\backslash\{\tau(K)\}$，不等式 $\Pr[\mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0))\neq\perp]\leqslant v(\eta)$ 成立，结合归纳假设有

$$1-(|E_0||T|v(\eta)+(|T|-1)v(\eta))\geqslant 1-(|E_0|+1)|T|v(\eta)$$
$$\geqslant 1-|E||T|v(\eta)$$

可以得到

$$\Pr[C_{kr}(e,\tau(T))\neq\{\tau(F_{kr}(E,T))\}]\leqslant|E||T|v(\eta)$$

引理 4.6 设 $\Pi=(\mathcal{K},\mathcal{E},\mathcal{D})$ 为一个无混淆的加密方案，$E\in\text{Exp}$，e 为根据分布 $[\![E]\!]_{\Pi(\eta)}$ 选

择的一个位串，τ 为密钥赋值，那么存在可忽略函数 $v(\cdot)$，使得

$$\Pr[\text{Crecoverable}(e)\neq\tau(\text{recoverable}(E))]\leqslant|E|^3 v(\eta)$$

证明 设 v 为与表达式 E 相关联的可忽略函数，通过对 E 的结构使用归纳法来证明该引理。考虑用于定义 recoverable 的集合 $\phi=G_0\subseteq G_1\subseteq\cdots$（即 $G_{i+1}=F_{\text{kr}}(E,G_i)$）和执行 Crecoverable 的过程中所计算出的集合 $\phi=T_0\subset T_1\subset\cdots$（即 $T_{i+1}=C_{\text{kr}}(e,T_i)$），如果用 A_i 表示第 i 次迭代事件，$T_i=\tau(G_i)$，注意到 A_0 为空，从而 A_0 的发生平凡地为真，有

$$\Pr[A_{i+1}]\geqslant\Pr[A_{i+1}\wedge A_i]=\Pr[A_{i+1}\mid A_i]\Pr[A_i]=(1-\Pr[\neg A_{i+1}\mid A_i])\Pr[A_i]$$

将引理 4.5 的结论应用于所有的 A_i，可得

$$\begin{aligned}&\Pr[T_{i+1}=\tau(G_{i+1})]\\=&(1-\Pr[C_{\text{kr}}(e,T_i)\neq\tau(F_{\text{kr}}(E,G_i))\mid T_i=\tau(G_i)])\Pr[T_i=\tau(G_i)]\\\geqslant&(1-|E||G_i|v(\eta))\Pr[T_i=\tau(G_i)]\\\geqslant&(1-|E|^2v(\eta))\Pr[T_i=\tau(G_i)]\end{aligned}$$

由于以上不等式中 i 的任意性，可将其替换为 $i-1,i-2,\cdots$，从而得到

$$\begin{aligned}\Pr[T_i=\tau(G_i)]&\geqslant(1-|E|^2v(\eta))^i\Pr[T_0=\tau(G_0)]\\&=(1-|E|^2v(\eta))^i\geqslant 1-i|E|^2v(\eta)\end{aligned}$$

由于 E 中所有密钥的数量显然是不大于 $|E|$ 的，且在每次迭代中至少有一个密钥被恢复（否则就停止迭代），从而有 $i\leqslant|E|$，因此 $\Pr[T_i\neq\tau(G_i(E))]\leqslant|E|^3v(\eta)$，由此立即可得引理的结论：

$$\Pr[\text{Crecoverable}(E)\neq\tau(\text{recoverable}(E))]\leqslant|E|^3v(\eta)$$

模式计算：已经证明，给定 $e\xleftarrow{R}[\![E]\!]_{\Pi(\eta)}$，可以以不可忽略的概率计算出赋给 recoverable(E) 中密钥的密钥值，下一步将证明在可计算模型中也存在一个与上文给出的函数 p 相对应的函数。

定义函数 psp:String$\times\mathcal{P}$(Key)$\rightarrow$Pattern，具体来说，函数 psp 可由以下算法定义：

```
Let f: tT→Keys be an injective function
Algorithm psp(e,tT)
    if e=⟨b,"block"⟩output block symbol B corresponding to b;
    if e=⟨k,"key"⟩and k∈tT then output f(k);
    if e=⟨(m,n),"pair"⟩output (psp(m,tT),psp(n,tT));
    if e=⟨c, "ciphertext"⟩
        if for exactly one key k ∈tT,D_k(c)≠⊥
        output {psp(D_k(c),tT)}_f(k)
        else output □;
```

函数的输入为一个串 e 和密钥值集合 tT，返回一个模式。算法使用一个任意的命名函数 f（一旦确定将固定下来），该函数对每一个密钥值关联一个唯一的密钥名，假设从分组的位串表示中得到其分组符号。下一个引理表明“语法”函数 p 与其相应的“计算”函数 psp 实质上具有相同的行为。

引理 4.7 设 Π 是一个无混淆的加密方案，$E\in$ Exp，$T\subseteq$Keys 为一个密钥符号集，e 为根据分布$[\![E]\!]_{\Pi(\eta)}$选择的一个位串，τ 为密钥赋值，那么存在可忽略函数 $v(\cdot)$，使得

$$\Pr[\mathrm{psp}(e,\tau(T)) \neq \mathrm{p}(E,T)] \leqslant |E||T|v(\eta)$$

证明 设 v 为与表达式 E 相关联的可忽略函数，函数 $\sigma: K \mapsto f(\tau(K))$ 为单射的密钥重命名函数，其中 f 为 psp 算法中所使用的命名函数，将通过对 E 的结构使用归纳法来证明以下结果是成立的：

$$\Pr[\mathrm{psp}(e,\tau(T)) \neq p(E,T)\sigma] \leqslant |E||T|v(\eta)$$

如果 $E=B$，则 $e=\langle b,\text{“block”}\rangle$，且 $p(E,T)=b$，其中 b 为 B 的位串描述，可以得到

$$\begin{aligned}\Pr[\mathrm{psp}(e,\tau(T)) \neq p(E,T)\sigma] &= 1-\Pr[\mathrm{psp}(e,\tau(T)) = p(E,T)\sigma] \\ &= 1-\Pr[b = b\sigma] = 0;\end{aligned}$$

如果存在 $K \in \mathrm{Keys}$ 使得 $E=K$，则 $e=\langle \tau(K),\text{“key”}\rangle$，且 $p(E,T)=K$，可以得到

$$\begin{aligned}\Pr[\mathrm{psp}(e,\tau(T)) \neq p(E,T)\sigma] &= 1-\Pr[f(\tau(K)) = K\sigma] \\ &= 1-\Pr[f(\tau(K)) = f(\tau(K))] = 0\end{aligned}$$

如果存在 $E_0,E_1 \in \mathrm{Exp}$ 使得 $E=(E_0,E_1)$，那么 $e=\langle(e_0,e_1),\text{“pair”}\rangle$，其中 $e_i \xleftarrow{R} [\![E_i]\!]_{\Pi(\eta)}$，$i=0,1$，且有 $p(E,T)=(p(E_0,T),p(E_1,T))$，那么有下式成立：

$$\begin{aligned}\Pr[\mathrm{psp}(e,\tau(T)) \neq p(E,T)\sigma] &\leqslant \Pr[\mathrm{psp}(e_0,\tau(T)) \neq p(E_0,T)\sigma \text{ or } \mathrm{psp}(e_1,\tau(T)) \\ &\qquad \neq p(E_1,T)\sigma] \\ &\leqslant \Pr[\mathrm{psp}(e_0,\tau(T)) \neq p(E_0,T)\sigma] \\ &\qquad + \Pr[\mathrm{psp}(e_1,\tau(T)) \neq p(E_1,T)\sigma]\end{aligned}$$

由归纳假设可得

$$\begin{aligned}\Pr[\mathrm{psp}(e,\tau(T)) \neq p(E,T)\sigma] &\leqslant |E_0||T|v(\eta) + |E_1||T|v(\eta) \\ &= (|E_0|+|E_1|)|T|v(\eta) = |E||T|v(\eta)\end{aligned}$$

如果 $E=\{E_0\}_K$，那么 $e=\langle \mathcal{E}_{\tau(K)}(e_0),\text{“ciphertext”}\rangle$，其中 $e_0 \xleftarrow{R} [\![E_0]\!]_{\Pi(\eta)}$。分两种情况：

(1) 如果 $K \notin T$，则 $p(E,T)=\square$，可以得到

$$\Pr[\mathrm{psp}(e,\tau(T)) = p(E,T)\sigma] = \Pr[(\forall k_i \in \tau(T))\ \mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0)) = \bot]$$

它蕴涵着

$$\begin{aligned}&\Pr[\mathrm{psp}(e,\tau(T)) \neq p(E,T)\sigma] \\ &= Pr[(\exists k_i \in \tau(T))\ \mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0)) \neq \bot] \\ &\leqslant \sum_{k_i \in \tau(T)} \Pr[\mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0))] \leqslant |T|v(\eta) \leqslant |E||T|v(\eta)\end{aligned}$$

(2) 如果 $K \in T$，$p(E,T)=\{p(E_0,T)\}_K$，从而下式成立：

$$\begin{aligned}&\Pr[\mathrm{psp}(e,\tau(T)) = p(E,T)\sigma] \\ &\geqslant \Pr[\mathrm{psp}(e_0,\tau(T)) = p(E_0,T)\sigma \text{ and } (\forall k_i \in \tau(T)\backslash\{\tau(K)\}) \\ &\qquad \mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0)) = \bot] \\ &= 1-\Pr[\mathrm{psp}(e_0,\tau(T)) \neq p(E_0,T)\sigma \text{ or} \\ &\qquad (\exists k_i \in \tau(T)\backslash\{\tau(K)\})\ \mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0)) \neq \bot] \\ &\geqslant 1-(\Pr[\mathrm{psp}(e_0,\tau(T)) \neq p(E_0,T)\sigma] \\ &\qquad + \Pr[(\exists k_i \in \tau(T)\backslash\{\tau(K)\})\ \mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0)) \neq \bot]) \\ &\geqslant 1-(\Pr[\mathrm{psp}(e_0,\tau(T)) \neq p(E_0,T)\sigma] + \sum_{k_i \in \tau(T)\backslash\{\tau(K)\}} \Pr[\mathcal{D}_{k_i}(\mathcal{E}_{\tau(K)}(e_0)) \neq \bot])\end{aligned}$$

$$\geqslant 1-(|E_0||T|v(\eta)+|T|v(\eta))$$

最后一个不等式可由归纳假设以及 Π 的无混淆性得到。由此立即可得

$$\Pr[\text{psp}(e,\tau(T))\neq p(E,T)\sigma]\leqslant|E_0||T|v(\eta)+|T|v(\eta)\leqslant|E||T|v(\eta)$$

psp(e,Crecoverable(e))$\not\cong p(E$,recoverable(E))的概率可以用以上引理计算出来,其结果是可忽略的。由此可得,给定 $e\overset{R}{\leftarrow}[\![E]\!]_{\Pi(\eta)}$ 可以以不可忽略的概率恢复出 E 的模式 psp(e,Crecoverable(e))。

引理 4.8　设 Π 是一个无混淆的加密方案,$E,E'\in\text{Exp}$ 为两个表达式,且有 pattern(E)$\not\cong$pattern(E')。如果 e 为从分布$[\![E]\!]_{\Pi(\eta)}$中抽样的一个位串,τ 为密钥赋值,那么存在可忽略函数 $v(\cdot)$,使得

$$\Pr[tT\leftarrow\text{Crecoverable}(e):\text{pattern}(E)\not\cong\text{psp}(e,tT)]\leqslant 2|E|^3v(\eta)\qquad(4\text{-}15)$$

及

$$\Pr[tT\leftarrow\text{Crecoverable}(e):\text{pattern}(E')\not\cong\text{psp}(e,tT)]\geqslant 1-2|E|^3v(\eta)\qquad(4\text{-}16)$$

成立。

证明　首先,注意到由式(4-15)立即可得式(4-16)。的确,由 pattern(E)$\not\cong$pattern(E')可得

$$\Pr[tT\leftarrow\text{Crecoverable}(e):\text{pattern}(E')\not\cong\text{psp}(e,tT)]$$
$$\geqslant\Pr[\text{pattern}(E)\cong\text{psp}(e,tT)]\geqslant 1-2|E|^3v(\eta)$$

剩下的就是证明式(4-15)为真。设 v 为与表达式 E 相关联的可忽略函数,有

$$\Pr[tT\leftarrow\text{Crecoverable}(e):\text{pattern}(E)\cong\text{psp}(e,tT)]$$
$$\geqslant\Pr[tT=\tau(\text{recoverable}(E))\wedge\text{pattern}(E)\cong\text{psp}(e,tT)]$$
$$=\Pr[p(E,\text{recoverable}(E))\cong\text{psp}(e,tT)\mid tT=\tau(\text{recoverable}(E))]$$
$$\cdot\Pr[tT=\tau(\text{recoverable}(E))]$$
$$=\Pr[p(E,\text{recoverable}(E))\cong\text{psp}(e,\tau(\text{recoverable}(E))]$$
$$\cdot\Pr[\text{Crecoverable}(e)=\tau(\text{recoverable}(E))]$$

分别用引理 4.7 和引理 4.8 对上式的两个因子取其界值,可得

$$\Pr[tT\leftarrow\text{Crecoverable}(e):\text{pattern}(E)\cong\text{psp}(e,tT)]$$
$$\geqslant(1-|E|^2v(\eta))\cdot(1-|E|^3v(\eta))\geqslant 1-2|E|^3v(\eta)$$

由此可得式(4-15)。

定理 4.6 的证明　对任意两个表达式 E_0 和 E_1,且 pattern(E_0)$\not\cong$pattern(E_1),可以证明,存在一个算法可以不可忽略的优势区分$[\![E_0]\!]_{\Pi(\eta)}$和$[\![E_1]\!]_{\Pi(\eta)}$。定义以下一个区分器 $D(e,\eta)$,其中 e 为$[\![E_0]\!]_{\Pi(\eta)}$或$[\![E_1]\!]_{\Pi(\eta)}$的一个抽样,η 为安全参数。计算模式 psp(e,Crecoverable(e)),并将其与 E_0 的模式相比较,这一点可由统一的算法在多项式时间内完成。如果它们在要重命名情况下是等价的,那么算法输出 0,否则输出 1。该算法的区分能力遵循引理 4.8 的式(4-15)和式(4-16)。更确切地说,算法 A 在区分$[\![E_0]\!]_{\Pi(\eta)}$和$[\![E_1]\!]_{\Pi(\eta)}$时的优势为

$$\Pr[e\overset{R}{\leftarrow}[\![E_0]\!]_{\Pi(\eta)}:D(e,\eta)=0]-\Pr[e\overset{R}{\leftarrow}[\![E_1]\!]_{\Pi(\eta)}:D(e,\eta)=0]$$
$$\geqslant(1-2|E_0|^3v(\eta))-2|E_1|^3v(\eta)=1-2(|E_0|^3+|E_1|^3)v(\eta)$$

其中,对任意长度为安全参数的多项式的表达式来说,$2(|E_0|^3+|E_1|^3)v(\eta)$是可忽略的。

形式化方法通常处理简单的、要么有要么无的安全断言,而计算方法中用到概率和计算

复杂性。直觉上,形式化断言在计算模型中如果不是绝对有效,也应当是在计算能力有限的敌手面前以很高的概率有效的。本节讨论了这一直觉,并证明了这一直觉在合理的假设下是正确的,即 AR 逻辑是合理的。进一步地,研究了 AR 逻辑的完备性,表明在一定的安全方案下它是不完备的,但如果给所讨论的加密方案附加一定的条件,则它也是完备的。

4.6 小结

本章重点介绍了 BAN 逻辑[1,2]、Kailar 逻辑[29]、Paulson 的定理证明系统[14]和 Blanchet 的自动验证系统,以及形式化方法的计算可靠性等内容。希望能够通过这些介绍,使读者掌握形式化分析和混合方法的基本分析思想和手段,为读者进一步研究提供必要的基础。

混合方法即把可证明安全性和形式化分析方法结合起来的方法是一个发展方向,本章重点介绍了形式化方法的计算可靠性方面的研究成果,主要取材于文献[18]和文献[19]。

关于形式化理论与方法方面的文献很多,除了前面已经提到的文献外,文献[31]~文献[48]也是值得一读的文献。本章在写作过程中得到了薛锐研究员的大力支持,他提供了大量的相关材料及他自己的一些认识,作者在此表示衷心的感谢。

参考文献

[1] Burrows M, Abadi M, Needham R. A logic of authentication. Rep. 39, Digital Equipment Corporation Systems Research Center, Palo Alto, Calif., Feb. 1989.

[2] Burrows M, Abadi M, Needham R. A logic of authentication. ACM Transactions on Computer Systems, vol. 8, 18~36, 1990.

[3] Dolev D, Yao A C. On the security of public key protocols0, Proceedings of the IEEE 22nd Annual Symposium on Foundations of Computer Science, 350~357, 1981.

[4] Dolev D, Even S, Karp R. On the Security of Ping-Pong Protocols. Information and Control, pages 57~68, 1982.

[5] Even S, Goldreich O. On the security of multi-party ping-pong protocols. In Proceedings of the 24th IEEE Symposium on the Foundations of Computer Science, 34 ~ 39. IEEE Computer Society Press, 1983.

[6] Millen J K, Clark S C, Freedman S B. The Interrogator: protocol security analysis. IEEE Transactions on Software Engineering, SE-13(2), 1987.

[7] Kemmerer R A. Analyzing encryption protocols using formal verifycation techniques, Advances in Cryptology -CRYPTO .87. Springer-Verlag LNCS 293 (1988). Editor: C. Pomerance. 289~305.

[8] Kemmerer R A. Analyzing encryption protocols using formal verification techniques. Advances in Cryptology -EUROCRYPT . 85, Linz, Austria. Springer-Verlag LNCS 219 (1986). Editor: F. Pichler. N. Koblitz, A. Menezes, Another look at "provable security" Technical Report CORR 2004-20, University of Waterloo, 2004. http://www.cacr.math.uwaterloo.ca/ ajmeneze/research.html.

[9] Vardi M Y. Why is modal logic so robustly decidable? In Descriptive Complexity and Finite Models, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 31, AMS, 1997, 149-184.

[10] Gavin Lowe. "Breaking and Fixing the Needham- Schroeder Public Key Protocol Using FDR", In Proceedings of TACAS, Vol. 1055 of Lecture Notes in Computer Science, 147~166, Springer-

Verlag, 1996.

[11] Abadi M. Secrecy by typing in security protocols. Journal of the ACM, 46 (5): 749 ~ 786, Sept. 1999.

[12] Gordon A, Jerey A. Authenticity by typing in security protocols. In Proceedings of the 14th IEEE Computer Security Foundations Workshop. IEEE Computer Society Press, June 2001.

[13] Abadi M, Gordon A D. A calculus for cryptographic protocols: The spi calculus. Information and Computation, 148(1):1~70, 1999.

[14] Paulson O, Quisquater J J. A security analysis of the Cliques protocols suites. In Proceedings of the 14th IEEE Computer Security Foundations Workshop, 73~81. IEEE Computer Society Press, June 2001.

[15] Fabrega F J T, Hertzog J, Guttman J. Strand spaces: Proving security protocols correct. Journal of Computer Security, 7(2/3):191~230, 1999.

[16] Dawn Xiaodong Song. "Athena: A New Efficient Automatic Checker for Security Protocol Analysis", 12th IEEE Computer Security Foundations Workshop, Jun 28~30, 1999.

[17] Blanchet B. Automatic verification of correspondences for security protocols. J. Comput. Secur. 17, 4 (Dec. 2009), 363~434.

[18] Abadi M, Rogaway P. Reconciling two views of cryptography: The computational soundness of formal encryption. In Proc. 1st IFIP International Conference on Theoretical Computer Science, volume 1872 of Lecture Notes in Computer Science, 3~22. Springer, 2000.

[19] Abadi M, Rogaway P. Reconciling two views of cryptography (the computational soundness of formal encryption). Journal of Cryptology, 5(2):103~127, Spring 2002.

[20] Micciancio D, Warinschi B. Completeness theorems for the Abadi-Rogaway logic of encrypted expressions. Journal of Computer Security, 12(1):99~129, 2004.

[21] Gligor V, Horvitz D O. Weak Key Authenticity and the Computational Completeness of Formal Encryption. In D. Boneh, editor, CRYPTO 2003, volume 2729 of LNCS, pages 530 ~ 547. Springer-Verlag, Aug. 2003.

[22] Micciancio D, Warinschi B. Soundness of Formal Encryption in the Presence of Active Adversaries. in Theory of Cryptography Conference (TCC), Cambridge, Massachusetts, volume 2951 of Lecture Notes in Computer Science, 133~151, February 19~21 2004.

[23] Impagliazzo R, Kapron B. Logics for reasoning about cryptographic constructions. In STOC. 03, 372~383, 2003.

[24] Backes M, Pfitzmann B, Waidner M. A universally composable cryptographic library. Available as Cryptology ePrint Archive, Report 2003/015.

[25] Hoare C A R. Communicating Sequential Processes. Prentice-Hall, 1985.

[26] Roscoe A W. Model-checking CSP. In A Classical Mind: Essays in Honor of C. A. R. Hoare, Prentice-Hall, 1994.

[27] Cervesato I, Durgin N, Lincoln P, Mitchell J, Scedrov A. A metanotation for protocol analysis. In Proceedings of 12th IEEE Computer Security Foundations Workshop. IEEE Computer Society Press, June 1999.

[28] Cervesato I, Durgin N, Lincoln P D. Relating strands and multiset rewriting for security protocol analysis. In Proc. 13th IEEE Computer Security Foundations Workshop, 35~51, 2000.

[29] Rajashekar Kailar. Accountability in electronic commerce protocols. IEEE Transactions on software engineering, 1996,22(5).

[30] Medvinsky G, Neuman B C. NetCash: A design for practical electronic currency on the Internet. In Proceedings of the ACM conference on Computer and Communication Security, November 1993.

[31] Bella G. Formal Correctness of Security Protocols. Springer, 2007.

[32] Canetti R. Universally composable security: A new paradigm for cryptographic protocols. In Proc. 42nd IEEE Symposium on Foundations of Computer Science (FOCS), 136～145, 2001.

[33] Ivan Damg ard and Jesper Buus Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In Moti Yung, editor, Advances in Cryptology, Proceedings of CRYPTO 2002, volume 2442 of Lecture Notes in Computer Science, 581～596. Springer-Verlag, 2002. Full version online available at http://eprint. iacr. org/2001/091.

[34] Lincoln P, Mitchell J, Mitchell M. Probabilistic polynomial-time equivalence and security analysis. Wing In J, Woodcock J, Davies J, editors, FM. 99- Formal Methods, 776～793. Springer-Verlag LNCS 1709, September 1999.

[35] Milner R (1980). A calculus of communication systems, LNCS 92, Springer-Verlag.

[36] Mitchell J, Ramanathan A, Scedrov A. A probabilistic polynomialtime calculus for analysis of cryptographic protocols (Preliminary report), 17-th Annual Conference on the Mathematical Foundations of Programming Semantics, Arhus, Denmark, May, 2001, Electronic Notes in Theoretical Computer Science, Volume 45 (2001).

[37] Millen J K, Clark S C, Freedman S B. The Interrogator: protocol security analysis. IEEE Transactions on Software Engineering, SE-13(2), 1987.

[38] Needham R M, Schroeder M D. Using Encryption for authentication in large networks of computers. Communications of the ACM, 21(12):993～999, December 1978.

[39] Lawrence C. Paulson. The inductive approach to verifying cryptographic protocols. Journal of Computer Security, 6:85～128, 1998.

[40] Pereira O, Quisquater J J. Generic Insecurity of Cliques-Type Authenticated Group Key Agreement Protocols. In Proceedings of the 17-th IEEE Computer Security Foundations Workshop, 16～29, IEEE Computer Society Press.

[41] Roscoe A W. Model-checking CSP. In A Classical Mind: Essays in Honor of C. A. R. Hoare, Prentice-Hall, 1994.

[42] Stubblebine S, Gligor V. On message integrity in cryptographic protocols. In Proceedings of the 1992 Symposium on Security and Privacy, 85～104. IEEE Computer Society Press, May 1992.

[43] Van Oorschot P C. Extending cryptographic logics of belief to key agreement protocols (extended abstract). In Proceedings of the First ACM Conference on Computer and Communications Security, 232～243. ACM, November 1993.

[44] 冯登国,范红.安全协议形式化分析理论与方法研究综述,中国科学院研究生院学报,Vo. 20,No. 4,2003,389～406.

[45] 薛锐,冯登国.安全协议的形式化分析技术与方法,计算机学报,Vo. 29,No. 1,2006,1～20.

[46] Qingguang Ji, Sihan Qing, Yongbin Zhou, Dengguo Feng. Study on Strand Space Model Theory. J. Comput. Sci. Technol. 18(5): 553～570 (2003).

[47] Rui Xue, Dengguo Feng. New Semantic Model for Authentication Protocols in ASMs. J. Comput. Sci. Technol. 19(4): 555～563 (2004).

[48] 季庆光,冯登国.对几类重要网络安全协议形式模型的分析,计算机学报,2005,28(07): 1071～1083.

第 5 章　零知识证明理论与方法

数学家们通常把证明写在纸上来宣称他们最新发现的定理。然而在许多密码学意义上的应用场合中，还需要向他人证明我们的确知道某一定理的证明，但不愿意透露这一证明本身。想象以下一个简单的身份证明。每个人的身份对应一个大的合数，当需要展示自己身份时，向对方证明自己知道这一合数的素因子。

如果直接把素因子写在纸上展示给对方，对方立即可以伪装成我们。显然没有人想这么做。在这种情况下，我们的目标是：

(1) 使对方确信我们的身份(拥有与身份对应的合数的素因子)。

(2) 在确认过程中，这些素因子没有被泄露。

这两个目标看起来自相矛盾，然而，它们却由 Goldwasser、Micali 和 Rocko 的天才想法同时实现了。从本质上讲，他们的想法源于对什么是证明这一根本问题深刻的洞察，把证明写在纸上只是证明的一种方式，也可以通过“问答”这一交互的方式来进行。如果能容忍证明中一些微小的错误，则交互使得同时满足(1)和(2)成为可能。

这就是本章要讨论的主题——零知识证明理论与方法。

零知识证明这一概念是由 Goldwasser 等人[1,2]于 20 世纪 80 年代初提出的。零知识证明是一种协议，这种协议的一方称为证明者，它试图使被称为验证者的另一方相信某个论断是正确的，却不向验证者提供任何有用的信息。Goldwasser 等人提出的零知识证明是交互式的，也就是证明者和验证者之间必须进行交互，才能实现零知识性，因而称为交互零知识证明。Blum 等人[3~5]于 20 世纪 80 年代末通过利用一个共同的称为参考串的短随机串代替交互实现了零知识证明这一思想，他们把这种零知识证明称为非交互零知识证明，这种证明是非交互的、单向的，也就是证明者和验证者在定理证明阶段无需进行交互，就能实现零知识性。非交互零知识证明比交互零知识证明的适用范围更广，因此大大地扩充了零知识证明思想的应用。

零知识证明在信息安全中处于极为重要的基础性地位。自从它诞生以来，特别是 Goldreich 等人证明任何 NP 语言都有一个关于其成员问题的计算零知识证明以后，零知识证明便成了一个重要而有力的工具，它为多方安全计算这一几乎所有密码学任务的通用解决方案提供了关键的工具。另外，零知识协议还被广泛地应用于大量特定的安全协议的设计中，如身份认证、电子现金、电子投票、群组签名等。

5.1　交互零知识证明理论与方法

在交互零知识证明(Interactive Zero Knowledge Proofs)的研究中，目前人们最关心的基本模型有两种：一种是 GMR 模型[1]，在这种模型中，证明者具有无限的计算能力，验证者具有多项式时间的计算能力，证明指的是语言成员问题，即输入 I 是否是语言 L 的一个成员。GMR 的零知识证明不是真正的零知识证明，这是因为在证明中，证明者向验证者揭

露了知识的 1b,即 $I \in L$。但除此之外,再没有其他任何附加的信息泄露给验证者,通常称这种交互零知识证明为成员或定理的零知识证明(Zero Knowledge Proofs of Membership or Theorem);另一种是 FFS 模型[6],在这种模型中,证明者和验证者均具有多项式时间的计算能力,证明者的目的不是向验证者证明 $I \in L$,而是证明他知道 I 关于 L 的状况。FFS 的零知识证明是真正的零知识证明,因为在证明中,验证者没有得到任何信息,他连 $I \in L$ 还是 $I \notin L$ 都不知道,但他相信这个证明,通常称这种交互零知识证明为知识或身份的零知识证明(Zero Knowledge Proofs of Knowledge or Identity)。

下面用一个例子来说明这两种交互零知识证明的差别。例如,若一个数学家解决了费马大定理,当他使用 GMR 的零知识证明时,他不仅能使验证者相信他解决了这个问题,而且能使验证者知道他是证明了这个问题还是否定了这个问题。当他使用 FFS 的零知识证明时,他只能使验证者相信他解决了这个问题,但验证者并不知道他证明了这个问题还是否定了这个问题。当然,这仅仅是一个例子,费马大定理已被证明是正确的。

为了便于理解零知识,引用文献[7]中关于洞穴的故事来解释这一概念。这个洞穴如图 5.1 所示,里面有一个秘密,知道咒语的那些人能打开 C 和 D 之间的秘密之门。对其他任何人来说,这两条路都是死胡同。

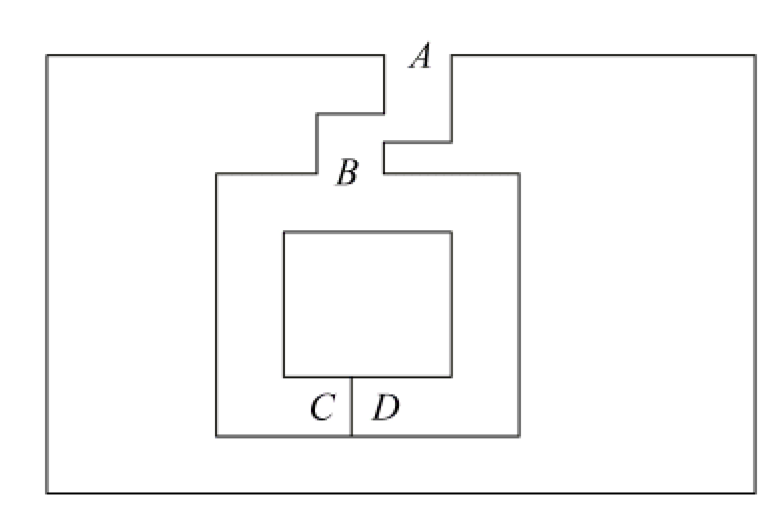


图 5.1　零知识洞穴

证明者 P 知道这个洞穴的秘密,他想让验证者 V 相信这一事实,但他不想泄露咒语。下面是 P 怎样使 V 相信的过程:

(1) V 先站在 A 点。

(2) P 走进洞穴,到达 C 点或 D 点。

(3) 在 P 消失在洞穴中之后,V 走到 B 点。

(4) V 向 P 喊,叫 P 或者从左通道出来,或者从右通道出来。

(5) P 答应了,如果有必要的话他就用咒语打开秘密之门。

(6) P 和 V 重复步骤(1)～(5)。

因为 P 没有办法重复猜出 V 要他从哪一边出来,所以如果 P 不知道这个秘密,那么他只能从进去的一边出来而不能从另一边出来。在每一轮中 P 有 $\frac{1}{2}$ 的机会猜中 V 会叫他从哪一边出来,所以有 $\frac{1}{2}$ 的机会愚弄 V。在两轮中 P 愚弄 V 的机会是 $\frac{1}{4}$。而在 n 轮后 P 愚弄 V 的机会是 $\frac{1}{2^n}$。当 $n=16$ 时,P 愚弄 V 的机会只有 $\frac{1}{65536}$。因此,如果所有 16 次 P 的证明都是对的,那么 V 可以相信 P 一定知道开启 C 点和 D 点间门的咒语。

5.1.1　成员的零知识证明

为了介绍成员的零知识证明这一概念,首先必须引入一系列其他概念。

1. 成员的交互证明系统

粗略地讲,成员的交互证明系统中的证明者 P 和验证者 V 可被形式化为一对交互的概

率图灵机，其中 V 限定为多项式时间概率图灵机，P 为计算能力无限的概率图灵机。证明系统(P,V)的公共输入为某个语言 L(如那些所有可以三着色的图的集合)的一个成员 x(如一个可以三着色的图)，P 的目的是向验证者 V 证明 $x\in L$(即 x 所代表的图可以三着色)。证明者和验证者按照预先的指令交替地向对方发送消息来进行这一证明过程。

一个交互图灵机(Interactive Turing Machine)是一个具有一条只读输入带、一条工作带、一条随机带、一条只读通信带和一条只写通信带的图灵机。随机带上包含一条无限长的随机比特序列，它只能从左到右地读入。可以说一个交互图灵机掷一个硬币意指它从自己的随机带上读取下一个比特。

一个交互协议(Interactive Protocol)是满足下列两个条件的一对有序图灵机(P,V)：P 和 V 共享同一条输入带；V 的只写通信带是 P 的只读通信带，反之，V 的只读通信带是 P 的只写通信带。机器 P 具有无限的计算能力，而机器 V 具有多项式时间的计算能力。所谓一个机器具有多项式时间的计算能力是指该机器关于任何输入所完成的全部计算或操作所需的时间都是它的输入长度的一个多项式。一个交互协议如图 5.2 所示，它的工作原理是，V 首先开始工作，然后两台机器轮流工作。在机器 P 或 V 的工作阶段，P 或 V 首先使用公共输入带和它的工作带、通信带、随机带上的内容完成某种内部计算；其次，P 或 V 在它的只写通信带上为 V 或 P 写一个串。P 或 V 的第 i 条消息是 P 或 V 在它的第 i 个工作阶段写在它的通信带上的全部串。一旦机器 P 或 V 写了它的消息，它就不工作，而机器 V 或 P 开始工作(除非协议被终止)。无论哪一台机器都能通过在一个工作阶段不发送任何消息来终止协议的计算。机器 V 通过输出接收(或拒绝)和终止协议来接收(或拒绝)公共输入。机器V 的计算时间是指 V 在各个工作阶段的计算时间的总和，这个时间是多项式的。

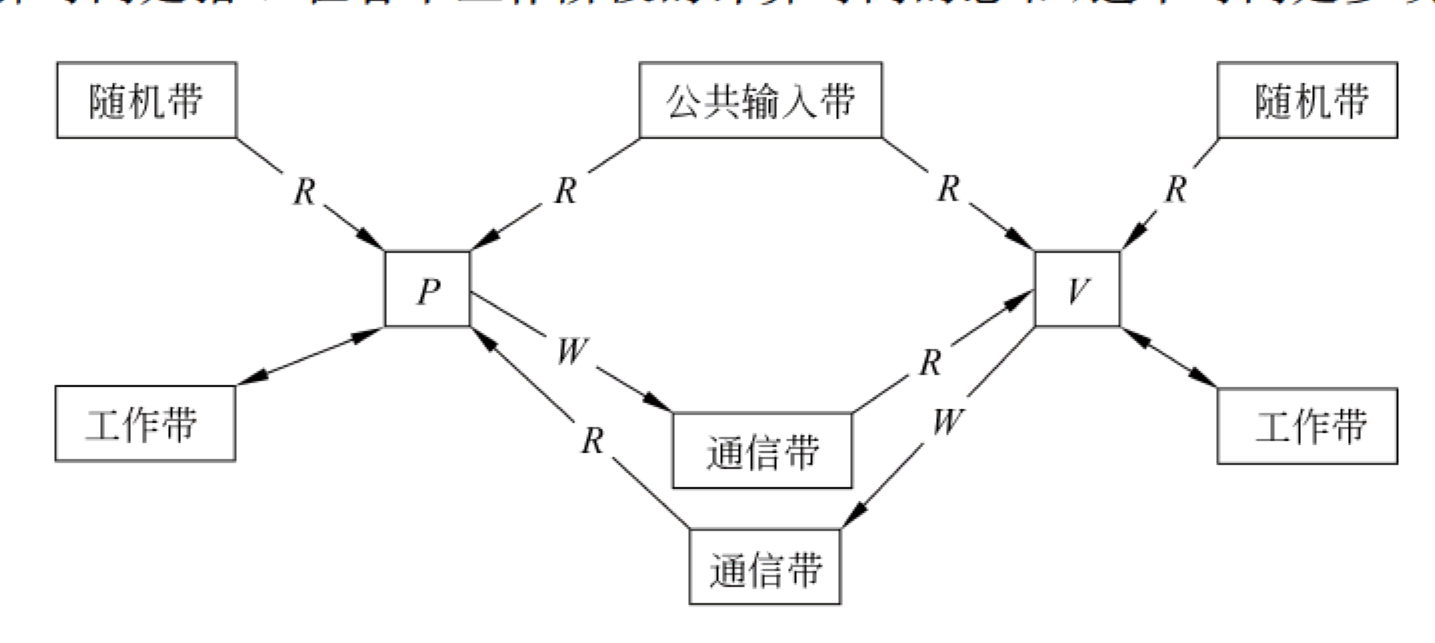


图 5.2　一个交互协议

图中，“↔”表示一个读、写头，“←R—”表示一个只读头，“—W→”表示一个只写头。

通常所说的“语言”是指一个集合 L。因为通常集合 L 中的每个元素 x 都可编码成一个 0、1 有限长串，该串的长度称为 x 的长度，记为 $|x|$，所以可以抽象地将 L 视作集合 $\{0,1\}^*$ 的一个子集。这里 $\{0,1\}^*$ 表示所有有限长的 0、1 串构成的集合。

设 $L\subset\{0,1\}^*$ 是一个语言，(P,V)是一个交互协议。我们说(P,V)对 L 是一个成员的交互证明系统(Interactive Proof System of Membership)，如果它满足下列两个条件：

(1) 完全性(Completeness)：对每一个 $k>0$ 和充分长的 $x\in L$，将 x 作为(P,V)的输入，V 终止协议并至少以 $1-|x|^{-k}$ 的概率接收 x。这里的概率是相对于协议(P,V)所有可能的掷硬币而言的。

(2) 合理性(Soundness):对每一个 $k>0$ 和充分长的 $x\notin L$,对任意的交互图灵机 P',将 x 作为 (P',V) 的输入,V 至多以 $|x|^{-k}$ 的概率接收 x。这里的概率是相对于协议 (P',V) 所有可能的掷硬币而言的。

注:在上述定义中,误差率 $|x|^{-k}$ 可以减弱为某常数 $0\leqslant\varepsilon<\frac{1}{2}$,也可加强为 $2^{-|x|}$,这 3 种误差率的交互证明系统是等价的[8]。通常将上述合理性定义中的概率 $\Pr[(P',V)(x)=接受]$ 称为合理性错误。

条件(1)本质上是说,如果 $x\in L$,P 能以很大的概率说服 V 接收 x。条件(2)是说,如果 $x\notin L$,无论 P' 采取何种欺骗策略,P' 说服 V 接收 x 的概率很小。事实上,V 无须相信正在与它进行交互的机器是否诚实,只要相信它自己掷硬币的随机性就足够了。条件(2)也表明,交互证明系统的定义依赖于 V,而根本不依赖于 P。在一个交互证明系统 (P,V) 中,通常将 P 称为证明者,V 称为验证者。

设 (P,V) 为一对交互图灵机。记 $(P(w),V(z))(x)$ 为图灵机 V 在停止交互后的输出,这里 x 为 P 和 V 的公共输入,w 和 z 分别是 P 的私有输入和 V 的辅助输入。有时为了方便,会忽略 P 或 V 的私有输入,把 $(P(w),V(z))(x)$ 记作 $(P(w),V)(x)$ 或 $(P,V)(x)$。注意到 P 和 V 一般都是概率图灵机(在交互过程中都将使用自己的随机带),可以把 $(P(w),V(z))(x)$ 看成一个随机变量。

下面将对二次剩余(Quadratic Residues)问题描述一个成员的交互证明系统。所谓二次剩余问题是指给定 $n\in N$(N 是全体自然数之集)和 $y\in Z_n^*=\{y\in Z_n\,|\,\gcd(y,n)=1\}$,确定 y 是否是一个模 n 的二次剩余。记为

$$\mathrm{QR}=\{(n,y)\mid n\in N, y\in Z_n^*, y\text{ 是模 } n \text{ 的一个二次剩余}\}$$

$$\mathrm{QNR}=\left\{(n,y)\mid n\in N, y\in Z_n^*, y\text{ 不是模 } n \text{ 的一个二次剩余,但}\left(\frac{y}{n}=1\right)\right\}$$

其中 $\left(\frac{y}{n}\right)$ 表示 y 模 n 的 Jacobi 符号,n 和 y 以二进制的形式给出。通常将 QR 称为二次剩余语言,QNR 称为二次非剩余(Quadratic Non-residues)语言。下面就对二次非剩余语言 QNR 来描述一个成员的交互证明系统。

设 (P,V) 是一个交互协议,(n,y) 是它们的公共输入,$m=|n|$,其中 $|n|$ 表示 n 的二进制表示的长度。V 检查是否 $n\geqslant1$,$y\in Z_n^*$,$\left(\frac{y}{n}\right)=1$,如果不是则停机,否则,$P$ 和 V 重复执行下列步骤 m 次。

(1) V 从它的随机带上读出一个整数 $i=0$ 或 1(将该整数称为挑战(Challenge))和一个整数 $r\in Z_n^*$,即 V 随机地选择一个整数 $i=0$ 或 1 和一个整数 $r\in Z_n^*$,计算 $w=y^ir^2 \bmod n$,并将 w 写在它的只写通信带上,即将 w 发送给 P。

(2) P 从它的只读通信带上读出 w,即 P 从 V 处收到 w 后,它验证是否 $(n,w)\in\mathrm{QR}$,如果 $(n,w)\in\mathrm{QR}$,那么 P 定义 $j=0$,否则它定义 $j=1$,然后它将 j 写在只写通信带上,即将 j 发送给 V(通常将 j 称为对挑战的响应(Response))。

(3) V 验证是否 $i=j$。

如果在 m 轮中的每一轮都有 $i=j$,那么 V 接收 P 的证明,即认为 $(n,y)\in\mathrm{QNR}$。

上述交互协议是一个交互证明系统。这是因为,如果 $(n,y)\in\mathrm{QNR}$,由 P 和 V 的执行过

程可知，在每一轮中都有 $i=j$，V 以概率 1 接收 P 的证明，即该协议是完全的。如果$(n,y)\notin$ QNR，则可假定$(n,y)\in$ QR，这是因为其他情况由 V 在开始时的检查可被排除或拒绝。此时无论 $i=0$ 还是 $i=1$，w 都是模 n 的二次剩余，P 无法确定是 $j=0$ 还是 $j=1$，它猜中的概率为$\frac{1}{2}$，这样只有当 m 轮中的每一轮中的 j 都猜中时，V 才接收，这种做法成功的概率只有$\frac{1}{2^m}$。因此，当$(n,y)\notin$ QNR 时，V 接收P'的证明的概率不超过$\frac{1}{2^m}$，即该协议是合理的。

再来看一下 V 的计算时间。数论知识说明，对任意给定的 $n\in N$ 和 $y\in Z_n$，利用欧几里德算法可在$|n|$的多项式时间内计算出是否 $y\in Z_n^*$，而且$\left(\frac{y}{n}\right)$也可在$|n|$的多项式时间内计算出。而在 V 的每一轮中，V 计算 $w=y^ir^2 \bmod n$ 和验证 $i=j$ 的时间都是多项式的，故 V 的计算时间是多项式的。

2. 随机变量的不可区分性和可逼近性

设 $L\subset\{0,1\}^*$，$U=\{U_{(x)}\}_{x\in L}$ 和 $V=\{V_{(x)}\}_{x\in L}$ 是两族随机变量，所有随机变量都在$\{0,1\}^*$中取值。我们想表达这样一个事实：随着 x 的长度的增加，$U_{(x)}$ 本质上可由 $V_{(x)}$ 来"取代"。现在考虑下述框架。

从 $U_{(x)}$ 或者从 $V_{(x)}$ 中抽取出一批随机样本并将这些随机样本交给一个判决者。判决者在研究这些样本之后，他将作出判决。如果样本来自 $U_{(x)}$，则判定为 0，如果样本来自 $V_{(x)}$，则判定为 1。如果随着 x 的长度的增加，任何判决者都无法作出判决，或者只能与 $U_{(x)}$ 和 $V_{(x)}$ 无关地胡乱随意地判决，那么就说 $U_{(x)}$ 本质上可由 $V_{(x)}$ 来"取代"，或者说 $U_{(x)}$ 和 $V_{(x)}$ 不可区分。在这个框架中有两个相关参数值得考虑，即样本的数目和判决者作出判决所需的时间。通过对这两个参数做不同的限制就会得到不同的随机变量不可区分的概念，目前最关心的不可区分的概念有 3 个，即相等（Equality）、统计不可区分（Statistical Indistinguishability）和计算不可区分（Computational Indistinguishablity）。下面分别来介绍这 3 个概念。

我们说两族随机变量$\{U_{(x)}\}_{x\in L}$和$\{V_{(x)}\}_{x\in L}$在语言 L 上是相等的，如果对每个充分长的 $x\in L$，$U_{(x)}$ 和 $V_{(x)}$ 的概率分布相等，即对每个 $\alpha\in\{0,1\}^*$，$P(U_{(x)}=\alpha)=P(V_{(x)}=\alpha)$。这时也称$\{U_{(x)}\}_{x\in L}$和$\{V_{(x)}\}_{x\in L}$完美不可区分（Perfect Indistinguishability）。也就是说，这两个带参数 x 的随机变量族，对充分长的 $x\in L$ 是相等的。

由定义可以看出，如果两族随机变量$\{U_{(x)}\}_{x\in L}$和$\{V_{(x)}\}_{x\in L}$在语言 L 上是相等的，那么对充分长的 $x\in L$，判决者即使具有无限的计算能力和拥有无穷多的样本也无法判定这些样本来自 $U_{(x)}$ 还是来自 $V_{(x)}$。

我们说两族随机变量$\{U_{(x)}\}_{x\in L}$和$\{V_{(x)}\}_{x\in L}$在语言 L 上是统计不可区分的，如果对任意常数 $c>0$ 和每个充分长的 $x\in L$，都有 $\sum\limits_{\alpha\in\{0,1\}^*}\left|P(U_{(x)}=\alpha)-P(V_{(x)}=\alpha)\right|<|x|^{-c}$。

由定义可以看出，如果两族随机变量$\{U_{(x)}\}_{x\in L}$和$\{V_{(x)}\}_{x\in L}$在语言 L 上是统计不可区分的，那么对充分长的 $x\in L$，对拥有多项式个样本和具有无限计算能力的判决者，他也基本上无法判定这些样本来自 $U_{(x)}$ 还是 $V_{(x)}$。

例 5.1 设 $U_{(x)}$ 对所有长度为$|x|$的串赋予相同的概率 $2^{-|x|}$，$V_{(x)}$ 对除了全 0 和全 1 的长度为$|x|$的串赋予相同的概率 $2^{-|x|}$，对长度为$|x|$的全 0 串和全 1 串分别赋予概率 0 和

$2^{-|x|+1}$，则$\{U_{(x)}\}_{x\in L}$和$\{V_{(x)}\}_{x\in L}$是语言$L=\{0.1\}^*$上统计不可区分的两族随机变量。

上面在给出完美不可区分和统计不可区分的定义时，实际上将判决者视作一个概率图灵机，即带有一条随机带的图灵机。按上述两个定义，在定义计算不可区分性时，可以将判决者视作一个多项式时间的概率图灵机来给出相应的定义。但这里将判决者视作一个多项式规模的电路族，这是因为通常认为这种电路族是一种可能比多项式时间的概率图灵机接收能力更强的计算装置，详细理由可参阅文献[2]。

设$C=\{C_x\}_{x\in L}$是一族布尔电路，C_x是输出仅为0或1的布尔电路，C_x的输入是以x为参数的随机变量，即C_x的输入是按参数x确定的随机变量分布的随机串。如果存在一个常数$e>0$，使得对所有的布尔电路$C_x\in C$至多有$|x|^e$个门(门包括与门、或门、非门等)，则称C为多项式规模的电路族。

为了把来自某一概率分布的样本输入多项式规模的电路，将只考虑多项式界随机变量族。所谓$U=\{U_{(x)}\}_{x\in L}$是一个多项式界随机变量族，意指存在一个常数$d>0$，使得对所有的随机变量$U_{(x)}\in U$只对长度不超过$|x|^d$的串分配正概率。

设$U=\{U_{(x)}\}_{x\in L}$和$V=\{V_{(x)}\}_{x\in L}$是两个多项式界随机变量族，$C=\{C_x\}_{x\in L}$是多项式规模的电路族，用$P(U,C,x)$表示按$U_{(x)}$分布的随机串作为输入，C_x输出1的概率。称U和V在语言L上是计算不可区分的，如果对任意常数$c>0$和每个充分长的$x\in L$，都有$|P(U,C,x)-P(V,C,x)|<|x|^{-c}$。

由定义易知，对于两个随机变量族$U=\{U_{(x)}\}_{x\in L}$和$V=\{V_{(x)}\}_{x\in L}$，如果它们在语言L上是完美不可区分的，那么它们在语言L上必定是统计不可区分的。现在来说明，对于两个多项式界随机变量族$U=\{U_{(x)}\}_{x\in L}$和$V=\{V_{(x)}\}_{x\in L}$，如果它们在语言L上是统计不可区分的，那么它们在语言L上必定是计算不可区分的。

设C_x是一个电路，S_x是使得C_x的输出为1的输入集。因为U和V是统计不可区分的，所以对任意常数$c>0$和每个充分长的$x\in L$，都有$\sum_{\alpha\in\{0,1\}^*}|P(U_{(x)}=\alpha)-P(V_{(x)}=\alpha)|<|x|^{-c}$。

而$|P(U_{(x)}\in S_x)-P(V_{(x)}\in S_x)|=\left|\sum_{\alpha\in S_x}P(U_{(x)}=\alpha)-\sum_{\alpha\in S_x}P(V_{(x)}=\alpha)\right|\leqslant\sum_{\alpha\in S_x}|P(U_{(x)}=\alpha)-P(V_{(x)}=\alpha)|\leqslant\sum_{\alpha\in\{0,1\}^*}|P(U_{(x)}=\alpha)-P(V_{(x)}=\alpha)|$，所以$|P(U_{(x)}\in S_x)-P(V_{(x)}\in S_x)|<|x|^{-c}$。又$P(U,C,x)=P(U_{(x)}\in S_x)$，$P(V,C,x)=P(V_{(x)}\in S_x)$，所以$|P(U,C,x)-P(V,C,x)|<|x|^{-c}$，故$U$和$V$在语言$L$上是计算不可区分的。

现在来定义随机变量的可逼近性(Approximability)。设M是一个关于输入x以概率1停机的概率图灵机，用$M(x)$来表示一个随机变量，该随机变量的概率分布为：对每一个串α，$P(M(x)=\alpha)=p(M$关于输入x输出$\alpha)$，即$P(M(x)=\alpha)$定义为M关于输入x输出α的概率。

设$L\subset\{0,1\}^*$，$U=\{U(x)\}_{x\in L}$是一族随机变量，称U在语言L上是完美(统计、计算)可逼近的，如果存在一个多项式时间的概率图灵机M，使得$\{M(x)\}_{x\in L}$和$\{U_{(x)}\}_{x\in L}$在L上是完美(统计、计算)不可区分的。

由定义可知，随机变量的可逼近性和随机变量的不可区分性密切相关，每一种不可区分性对应一种可逼近性。如果U在L上是完美逼近的，那么U在L上必是计算可逼近的。当然在谈论随机变量的计算不可区分性和计算可逼近性时，是指多项式界随机变量的计算

不可区分性和计算可逼近性。

3. 成员的零知识证明

非正式地讲，一个成员的交互证明系统(P,V)是由 P 和 V 两方所执行的一个协议。P 试图通过执行协议，说服验证者 V 相信某个定理 T(如 $x\in L$)是正确的。如果 T 为假(如 $x\notin L$)，则即使 P 不遵循协议，采取任何欺骗策略，也无法说服 V 相信T 为真。但在一个成员的交互证明系统中，如果一个有欺骗行为的验证者 V'不遵循协议，他试图从 P 那里得到除了 T 为真以外的其他信息，即 V'通过协议可能获得了他不应得到的知识。现在来考虑即使 V'不遵循协议，采取任何欺骗策略，也无法从 P 那里得到除了 T 为真以外的任何其他信息的交互协议和交互证明系统。

首先描述一个欺骗验证者(Cheating Verifier)V'，它预先拥有某些额外信息且不遵循协议。

设(P,V)是一个交互协议，V'是一个带有一条额外输入带的交互图灵机。当公共输入为 x 时，V'的输入为(x,H)，H 是 V'的额外输入，且 H 的长度不超过$|x|$的多项式。当 V'与 P 交互时，P 只看到它的输入带上的 x，而 V'看到的是(x,H)，此时假定 V'的所有计算时间不超过$|x|$的多项式。H 的一个最好的解释是把它解释为欺骗验证者 V'已经拥有的关于 x 的一些知识或欺骗验证者在执行协议(P,V')之前极力使用从 P 得到的知识执行其他协议所获得的交互历史。

对关于公共输入为 x 和额外输入为 H 的一段协议，定义 V'的观察(View)是V'看到的一切事情。设 σ 和 ρ 分别是包含在 P 和 V'的随机带中的两个串，不妨设P 和 V'关于这些随机选择的计算由 n 轮构成，并且 V'首先开始工作，这里 a_i 和 b_i 分别是 P 和 V'的第 i 个消息。那么就说$(\rho,b_1,a_1,\cdots,b_n,a_n)$是 V'关于输入(x,H)的观察。设 $\mathrm{View}_{P,V'}(x,H)$是一个随机变量，其值是这个观察。为方便起见，把每个观察视作来自$\{0.1\}^*$的长度不超过$|x|$的多项式的一个串。注意，可以把 V'的工作带中的内容包括在 $\mathrm{View}_{P,V'}(x,H)$中，也可以排除在 $\mathrm{View}_{P,V'}(x,H)$之外，这都无本质差别，因为它们都能在多项式时间内从 $x,H,\rho,a_1,a_2,\cdots,a_n$ 求得。

一个交互副本(Transcript)包括公共输入 x、P 和 V'在一次协议的执行过程中产生的所有消息。验证者 V'在停止交互后的输出可以有两种不同的形式：①仅输出“接受”(有时用“1”表示)或“拒绝”(有时用“0”表示)；②输出它的观察。事实上，对于定义零知识性而言，这两种形式是等价的。我们称一个副本是可接受的，是指验证者接受了这个副本中每一个来自证明者的消息。在某些上下文已交代清楚的情况下，为简单起见，也直接把 P 和 V'在协议的一次执行中产生的所有消息称为副本，而忽略了公共输入 x。应特别注意的是，任何副本都是针对于某个特定公共输入的副本。

设 $L\subset\{0,1\}^*$是一个语言，(P,V)是一个交互协议，V'是以上所描述的欺骗验证者。称(P,V)对 V'关于语言 L 是完美(统计、计算)零知识的，如果随机变量族 $\mathrm{View}_{P,V'}=\{\mathrm{View}_{P,V'}(x,H)\}_{(x,H)\in L'}$关于语言 $L'=\{(x,H)\mid x\in L,|H|$不超过$|x|$的多项式$\}$是完美(统计、计算)可逼近的。我们说$(P,V)$关于语言 L 是完美(统计、计算)零知识的，如果它是对所有的多项式时间的概率交互图灵机 V'关于语言 L 是完美(统计、计算)零知识的。

由定义可知，如果(P,V)关于语言 L 是完美(统计、计算)零知识的，那么所有的多项式

时间的概率交互图灵机，无论采用何种欺骗措施，都无法通过和 P 交互抽取关于 L 的成员的一些附加信息。这表明该定义只依赖于 P，而根本不依赖于 V。

零知识的定义中考虑了两种概率分布：①一种是由一个概率多项式时间验证者 V' 在与证明者 P 交互之后产生的概率分布；②另一种是由一个概率多项式时间机器 M 关于输入产生的概率分布，M 没有和任何证明者交互。零知识意味着对每个类型①的分布存在一个类型②的分布，使得这两个分布是“本质上相等的”。直观地说，零知识意味着 V 与 P 交互所得到的信息是 V 单独也能得到的。有时将②中的 M 称为伪造算法（Forging Algorithm）或模拟器（Simulator）。

下面就来定义成员的零知识证明系统这一概念。

设 $L\subset\{0,1\}^*$ 是一个语言，(P,V) 是一个交互协议，我们说 (P,V) 对语言 L 是一个成员的完美（统计、计算）零知识证明系统，如果它对 L 是一个成员的交互证明系统并且关于 L 是一个完美（统计、计算）零知识协议。

由定义易知，如果 (P,V) 对语言 L 是一个成员的完美零知识证明系统，那么 (P,V) 对语言 L 一定是一个成员的统计零知识证明系统。如果 (P,V) 对语言 L 是一个成员的统计零知识证明系统，那么 (P,V) 对语言 L 一定是一个成员的计算零知识证明系统。

通常将计算零知识证明系统称为零知识证明系统，简称零知识证明。

如果 (P,V) 对 L 是一个成员的零知识证明系统，那么所有的概率多项式时间交互图灵机，无论采用何种欺骗措施，都不可能通过和 P 交互抽取除了 $x\in L$ 之外的别的信息。

前面已对二次非剩余语言 QNR 描述了一个成员的交互证明系统，现在来考察一下该系统的零知识性。

假定有一个知道 P 和 V 的公共输入 $x=(n,y)\in$ QNR 的第三方 C，C 随机地选择 $\tilde{r}\in Z_n^*$ 和 $\tilde{i}\in\{0,1\}$，并计算 $\tilde{w}=y^{\tilde{i}}\tilde{r}^2 \bmod n$。$C$ 将 $\tilde{w}$ 给 V，但 V 并不知道 $\tilde{w}$ 是否是一个二次剩余。V 在第(2)步将 $\tilde{w}$ 发送给 P，P 在第(3)步对 V 作出回答。这样 V 就借助 P 能够回答他事先不知道的并且有可能不能解决的问题。直观上看，该系统不是零知识的。这里不再作详细讨论，文献[2]中证明了该系统是统计零知识证明系统。

本节最后对二次剩余语言 QR 描述一个成员的零知识证明系统。

设 (P,V) 是一个交互协议，(n,y) 是它们的公共输入，$m=|n|$，其中 $|n|$ 表示 n 的二进制表示的长度。V 检查是否 $x\geqslant 1$，$y\in Z_n^*$，如果不是则停机，否则，P 和 V 重复执行下列步骤 m 次。

(1) P 从它的随机带上读出一个整数 $r\in Z_n^*$，即 P 随机地选择一个整数 $r\in Z_n^*$，计算 $w=r^2 \bmod n$，并将 w 写在它的只写通信带上，即将 w 发送给 V。

(2) V 从它的随机带上读出一个整数 $i=0$ 或 1，即 V 随机地选择一个整数 $i=0$ 或 1，并将 i 写在它的只写通信带上，即将 i 发送给 P。

(3) P 从它的只读通信带上读出 i，即 P 从 V 处收到 i 后，它计算 $Z=u^i r \bmod n$，这里 u 是 y 的一个平方根，即 $y=u^2 \bmod n$，并将 Z 写在它的只写通信带上，即将 Z 发送给 V。

(4) V 从它的只读通信带上读出 Z，即 V 从 P 处收到 Z 后，它验证是否 $Z^2\equiv y^i w \bmod n$。如果在 m 中的每一轮都有 $Z^2\equiv y^i w \bmod n$，那么 V 接收 P 的证明，即认为 $(n,y)\in$ QR。类似于 QNR 的情形，容易说明 V 的所有计算时间都不超过 $|n|$ 的多项式。

下面就来说明上述系统对语言 QR 是一个成员的完美零知识证明系统。

(1) 完全性：如果 $(n,y)\in QR$，由 P 和 V 的执行过程知，在每一轮中都有 $Z^2\equiv y^i w \bmod n$，所以 V 以概率 1 接收 P 的证明。

(2) 合理性：如果 $(n,y)\notin QR$，则可假定 $n\geqslant 1$，$y\in Z_n^*$，y 不是模 n 的一个二次剩余，这是因为其他情况由 V 在开始时的检查可被排除或拒绝。在每一轮中，V 从 P' 处收到的 w 不可能使 w 和 yw 都有模 n 的平方根。因为 P' 事先没有看到 V 随机选择的口令 i，他只好去猜，猜中的概率为 $\frac{1}{2}$，所以在每一轮中，V 接收 P' 的证明的概率为 $\frac{1}{2}$，这样只有当 P' 在 m 轮中的每一轮中都能正确地猜中口令 i 时，V 才接收 P' 的证明，这种做法成功的概率只有 $\frac{1}{2^m}$，因此 V 至多以概率 $\frac{1}{2^m}$ 接收 P' 的证明。

上述两个性质表明，协议 (P,V) 对 QR 是一个成员的交互证明系统。下面我们来说明该协议是(完美)零知识的。

(完美)零知识性：设 V' 是和 P 交互的任一个多项式时间交互图灵机，$(n,y)\in QR$ 是 (P,V') 的公共输入，$m=|n|$，$|n|$ 表示 n 的二进制表示的长度，H 是 V' 的额外输入，H 也可能是一个空串，V' 关于输入 $((n,y),H)$ 的观察值为 $((w_1,i_1,Z_1);(w_2,i_2,Z_2);\cdots;(w_m,i_m,Z_m))$，其中 $(w_j,i_j,Z_j)(1\leqslant j\leqslant m)$ 是 V' 关于输入 $((n,y),H)$ 在第 j 轮的观察值，w_j 是 P 随机选择的一个模 n 的二次剩余，i_j 是 V' 随机选择的一个整数(0 或 1)，Z_j 是 P 发送给 V' 的 $y^{i_j}w_j$ 模 n 的一个平方根。设 $\mathrm{View}_{P,V'}((n,y),H)=((w_1,i_1,Z_1);(w_2,i_2,Z_2);\cdots;(w_m,i_m,Z_m))$ 是一个随机变量，其值是 V' 关于输入 $((n,y),H)$ 的观察值。V' 可能遵循协议，也可能不遵循协议。

为了证明上述交互证明系统是完美零知识的，需对每一个 V'(诚实的或不诚实的验证者)，构造出一个相应的概率多项式时间图灵机 $M_{V'}$，使得 $M_{V'}((n,y),H)=\mathrm{View}_{P,V'}((n,y),H)$。$M_{V'}$ 扮演 P 的角色，并将 V' 用作一个可重新启动的子程序。$M_{V'}$ 极力猜测 V' 将在第 j 轮中选择的 i_j。也就是说，在第 j 轮中 $M_{V'}$ 首先产生一个形式为 (w_j,i_j,Z_j) 的随机的三元组，其中 $Z_j^2\equiv y^{i_j}w_j \bmod n$，然后运行机器 V'，看 V' 选择的口令 i_j'。如果 $M_{V'}$ 的猜测 i_j 和 V' 产生的 i_j' 相等，那么将 (w_j,i_j,Z_j) 级联到伪造的观察之后。否则，将这个三元组删掉，$M_{V'}$ 猜测一个新的口令 i_j 并将 V' 的状态调回原状态重新启动 V'。这里的"状态"是指由机器使用的所有变量的值。比方说，令 $V_j=((w_1,i_1,Z_1);(w_2,i_2,Z_2);\cdots;(w_j,i_j,Z_j))$ 是一个随机变量，在 V' 执行第 $j+1$ 轮时，V_j 已取定一个值 v_j，将 v_j 称为 V' 在执行 $j+1$ 轮时的一个状态，简称状态，记为 $S_j(V')$。约定 $S_0(V')=$ 空串。

对 V'，现在来详细描述一个多项式时间的概率图灵机 $M_{V'}$，$M_{V'}$ 的输入为 $(n,y)\in QR$ 和串。用 T 表示由 $M_{V'}$ 伪造的观察，置 $T=$ 空串。$M_{V'}$ 对 $j=1$ 到 m 执行下列步骤。

(1) $S_{j-1}(V')=T$。

(2) $M_{V'}$ 从它的随机带上读出一个整数 $i_j=0$ 或 1 和一个整数 $Z_j\in Z_n^*$，即 $M_{V'}$ 随机选择一个整数 $i_j=0$ 或 1 和一个整数 $Z_j\in Z_n^*$，并计算 $w_j=Z_j^2y^{-i_j} \bmod n$。

(3) 调用 V' 的第 j 轮，V' 的前 $j-1$ 轮的观察已由(1)中赋值。给 V' 输入 w_j，获得一个整数 i_j'，如果 $i_j=i_j'$，那么将 (w_j,i_j,Z_j) 级联到 T 的尾部，否则 $M_{V'}$ 返回步骤(2)，直到 $i_j=i_j'$ 为止。

首先来看一下 $M_{V'}$ 的计算时间。$M_{V'}$ 随机选择 $Z\in Z_n^*$ 的方法如下：$M_{V'}$ 首先随机选择一个长度为 m 的比特串，检测看它是否属于 Z_n^*，检测 Z 是否属于 Z_n^* 可在 $m=|n|$ 的多项式时间内完成，如果不是，$M_{V'}$ 随机选择一个长度为 $m-1$ 的比特串，检测看它是否属于 Z_n^*，依次类推，直到 $M_{V'}$ 随机选择一个长度为 1 的比特串进行检测为止，平均来说，这 m 个串中有 Z_n^* 中的元素，故平均来说，$M_{V'}$ 随机选择 $Z\in Z_n^*$ 可在 $m=|n|$ 的多项式时间内完成。

再看一下猜到 $i_j=i'_j$ 的平均运行时间有多长。不管 V' 怎样产生它的口令 i'_j，$M_{V'}$ 猜测 i_j 和 i'_j 一样的概率是 $\frac{1}{2}$。因此，平均来说，对每一个级联到伪造的观察尾部的三元组，$M_{V'}$ 将产生两个三元组。故平均运行时间是 $m=|n|$ 的多项式。可见，$M_{V'}$ 是一个概率多项式时间图灵机。

下面用数学归纳法来证明 $M_{V'}((n,y),H)=\text{View}_{P,V'}((n,y),H)$。对每一个 $j:0\leqslant j\leqslant m$，令 T_j 表示第 j 轮末的部分观察 $((w_1,i_1,Z_1);(w_2,i_2,Z_2);\cdots;(w_j,i_j,Z_j))$ 的全体集，设 $M_{V'}$ 在 T_j 上的概率分布为 $P_{M_{V'},j}$，$\text{View}_{P,V'}$ 在 T_j 上的概率分布为 $P_{\text{View}_{P,V'},j}$。注意到 $P_{M_{V'},m}=P_{M_{V'}}$，$P_{\text{View}_{P,V'},m}=P_{\text{View}_{P,V'}}$，因此，如果能证明对所有的 $j:0\leqslant j\leqslant m$，在 T_j 上的两个概率分布 $P_{M_{V'},j}$ 和 $P_{\text{View}_{P,V'},j}$ 是相等的，那么就证明了 $M_{V'}((n,y),H)=\text{View}_{P,V'}((n,y),H)$。

$j=0$ 对应的是机器的开始，此时 T_0 是空集，因此在 T_0 上的两个概率分布显然是相等的。

假定对某一 $j\geqslant 1$，在 T_{j-1} 上的两个概率分布 $P_{M_{V'},j-1}$ 和 $P_{\text{View}_{P,V'},j-1}$ 是相等的，现在证明在 T_j 上的两个概率分布 $P_{M_{V'},j}$ 和 $P_{\text{View}_{P,V'},j}$ 是相等的。

考虑交互证明中的第 j 轮。V' 随机选择整数 $i_j=0$ 的概率是某一实数 P_1，因此，随机选择整数 $i_j=1$ 的概率是 $1-P_1$，这里 P_1 依赖于机器 V' 的第 j 轮的状态。在交互证明中注意到，每一个 w 都由证明者 P 等可能地选择，又因为无论对哪一个可能的口令 i_j，所有的平方根都是等可能的，所以每一个 Z 都独立于 P_1 等可能地出现。因此，观察的第 j 个三元组是 (w,i,z) 的概率是 $\frac{P_1}{t}(i=0\text{ 时})$ 或 $\frac{1-P_1}{t}(i=1\text{ 时})$，这里 t 是 Z_n^* 中平方根的个数。

下面对 $M_{V'}$ 作一个类似的分析。在 $M_{V'}$ 的每一轮中的每一次迭代中，$M_{V'}$ 以概率 $\frac{1}{t}$ 选择任何平方根 Z。$i=0$ 并且 V' 的口令 $i'=0$ 的概率是 $\frac{P_1}{2}$；$i=1$ 并且 V' 的口令 $i'=1$ 的概率是 $\frac{1-P_1}{2}$。在这些情况中的每一种情况，(w,i,Z) 都能被作为伪造的观察的第 j 个三元组。在每一轮中的每一次迭代中，以 $\frac{1}{2}$ 的概率在带上没写任何东西。对 $i=0$ 的情形，在第 j 轮中，一个三元组 $(w,0,Z)$ 在 l 次迭代后被作为伪造的观察的第 j 个的概率是 $\frac{P_1}{2^l\times t}$。因此，$(w,0,Z)$ 是伪造的观察的第 j 个三元组的概率是 $\frac{P_1}{2\times t}+\frac{P_1}{2^2\times t}+\frac{P_1}{2^3\times t}+\cdots=\frac{P_1}{2\times t}\left(1+\frac{1}{2}+\frac{1}{4}+\cdots\right)=\frac{P_1}{t}$。类似于 $i=0$ 的情形，对 $i=1$ 的情形进行分析可得，$(w,1,Z)$ 是伪造的观察的第 j 个三元组的概率是 $\frac{1-P_1}{t}$。因此，在第 j 轮末，在 T_j 上的两个概率分布分

别为 $P_{M_{V'},j}=P_{M_{V'},j-1}\times\frac{P_1}{t}(i=0\text{ 时})$或 $P_{M_{V'},j-1}\times\frac{1-P_1}{t}(i=1\text{ 时})$和 $P_{\text{View}_{P,V'},j}=P_{\text{View}_{P,V'},j-1}\times\frac{P_1}{t}$ $(i=0\text{ 时})$或 $P_{\text{View}_{P,V'},j-1}\times\frac{1-P_1}{t}(i=1\text{ 时})$，由归纳假设知，$P_{\text{View}_{P,V'},j-1}=P_{M_{V'},j-1}$，故在 T_j 上的两个概率分布是相等的。由归纳法知，两个概率分布 $P_{M_{V'}}$ 和 $P_{\text{View}_{P,V'}}$ 是相等的。

现在针对子群成员(Subgroup Membership)问题给出另一个成员的完美零知识证明系统的例子。所谓子群成员问题是指给定 $n,l\in N$ 和 $\alpha,\beta\in Z_n^*$，$\alpha\neq\beta$，α 在 Z_n^* 中的阶为 l，则 $\langle\alpha\rangle=\{1,\alpha,\alpha^2,\cdots,\alpha^{l-1}\}$ 是由 α 生成的 Z_n^* 的一个 l 阶子群，确定 β 是否属于 $\langle\alpha\rangle$。记 $\text{SM}=\{(n,l,\alpha,\beta)\mid n,l\in N,\alpha,\beta\in Z_n^*,\alpha\text{ 的阶是 } l,\alpha\neq\beta,\beta\in\langle\alpha\rangle\}$，$n$、$l$、$\alpha$ 和 β 都以二进制的形式给出。将 SM 称为子群成员语言。下面针对 SM 语言来描述一个成员的完美零知识证明系统，这个协议的分析与对 QR 语言的完美零知识证明系统的分析类似，这里不再详细分析，感兴趣的读者可自己给出分析过程。

设 (P,V) 是一个交互协议，$(n,l,\alpha,\beta)\in\text{SM}$ 是它们的公共输入，$m=|n|$，$|n|$ 表示 n 的二进制表示的长度，V 检查是否 $n\geqslant1$，$l\geqslant1$，$\alpha,\beta\in Z_n^*$，$\alpha\neq\beta$，$\alpha^l=1$，如果不是则停机，否则，P 和 V 重复执行下列步骤 m 次。

(1) P 随机选择一个整数 $j\in Z_l$，计算 $\gamma=\alpha^j \bmod n$，并将 γ 发送给 V。

(2) V 随机选择一个整数 $i=0$ 或 1，并将 i 发送给 P。

(3) P 计算 $h=(j+ik) \bmod n$，这里 $k=\log_\alpha^\beta$，并将 h 发送给 V。

(4) V 验证是否 $\alpha^h\equiv\beta^i\gamma \bmod n$。

如果在 m 轮中的每一轮都有 $\alpha^h\equiv\beta^i\gamma \bmod n$，那么接收 P 的证明，即认为 $(n,l,\alpha,\beta)\in\text{SM}$。

前面给出的协议 (P,V) 都要求 P 和 V 之间依次执行若干轮，也可以要求 P 和 V 并行地一次把这些轮全部完成。前者通常称为串行协议，对应的零知识证明称为串行零知识证明。后者通常称为并行协议，对应的零知识证明称为并行零知识证明。二者之间的差别参见文献[9]。本小节讨论的证明系统中，只有一个证明者 P，也可以将此推广到多个证明者的情况，即证明者不只一个，详见文献[10]。关于成员的零知识证明就谈这些，感兴趣的读者可参见文献[2,10～13]。在这里值得一提的是，成员的零知识证明是针对某种语言而言的，究竟哪些语言具有成员的零知识证明还不太清楚。目前已证明[12]，如果单向置换存在，那么每一个 NP 问题都存在一个(计算)零知识证明系统。人们认为存在零知识证明系统的语言类要比 NP 类稍稍大一点。但很快人们就发现了一个 Co-NP 的语言(即图的非同构 GNI)也有一个交互证明，最终，交互的能力被发现远远超过了人们的想象，它所能刻画的语言类等于 PSPACE。

5.1.2 成员的零知识论证系统

交互证明系统有一种在密码学和安全协议中应用性极强的变体——交互论证系统(Interactive Argument System)。与交互证明系统相比，它稍微放宽了合理性条件，即只要求任意的概率多项式时间的证明者(而不是任意的有无限计算能力的证明者)不能欺骗诚实的验证者。

设 $L\subset\{0,1\}^*$ 是一个语言，(P,V) 是一个交互协议，P 和 V 均是概率多项式时间图灵

机。称(P,V)对L是一个成员的交互论证系统(Interactive Argument System of Membership),如果它满足下列两个条件:

(1) 完全性(Completeness):对每一个$k>0$和充分长的$x\in L$,将x作为(P,V)的输入,V终止协议并至少以$1-|x|^{-k}$的概率接收x。这里的概率是相对于协议(P,V)所有可能的掷硬币而言的。

(2) 合理性(Soundness):对每一个$k>0$和充分长的$x\notin L$,对任意的概率多项式时间交互图灵机P',将x作为(P',V)的输入,V至多以$|x|^{-k}$的概率接收x。这里的概率是相对于协议(P',V)所有可能的掷硬币而言的。

零知识论证系统的定义类似于零知识证明系统。从安全性的角度来讲,证明/论证系统的定义只涉及保护验证者不受欺骗(合理性)这一问题。零知识性是一个与证明者的安全性相关的概念。这一概念要求对于验证者而言,在证明结束时他除了相信对方所证明的断言为真外,没有获得任何额外的"知识"(或计算能力),他在证明当中所观察到的一切都可以被一个无须与证明者交互的概率多项式机器(即模拟器,Simulator)所模拟。

设$L\subset\{0,1\}^*$是一个语言,(P,V)是一个交互协议,我们说(P,V)对语言L是一个成员的完美(统计、计算)零知识论证系统,如果它对L是一个成员的交互论证系统并且关于L是一个完美(统计、计算)零知识协议。

如果证明/论证系统(P,V)中的所有验证者消息(除了它最后的输出)都是独立(与交互历史无关)的随机串,则称(P,V)为公开掷币的证明/论证系统。

在零知识的定义中,模拟器S的输入包括了验证者V'的代码,这也就是一般零知识的定义。如果模拟器S在计算过程中使用了验证者V'的代码,则称此证明/论证系统是非黑盒零知识的;如果模拟器S在计算过程中仅以黑盒(Black-box)的方式调用V',则称相应的零知识为黑盒零知识。一些研究表明,黑盒模拟确实是对零知识证明/论证系统的一种限制,例如,如果只利用黑盒模拟,则不可能构造出对于非平凡语言的常数轮公开掷币的零知识论证系统,但利用非黑盒模拟做到了这一点。就目前技术条件而言,黑盒零知识论证系统从效率上讲要比非黑盒零知识论证系统优越得多。在本章所涉及的构造中,绝大部分为黑盒零知识。

在一些上下文背景清晰的场合下,有时把零知识证明和论证系统统称为零知识协议。

5.1.3 对 NP 语言的零知识证明系统

对语言L,如果存在一个布尔关系$R_L=\{0,1\}^*\times\{0,1\}^*$和一个正多项式$p(n)$使得$R_L$能够在确定的多项式时间内识别,并且$x\in L$当且仅当存在$w$使得$|w|\leqslant p(|x|)$,$(x,w)\in R_L$,那么$L$属于 NP。通常称$w$为$x\in L$的证据。

如果L是一个 NP 语言,则当$x\in L$时,断言"$x\in L$"有一个小于多项式长度的"证明"(即证据w)。经常把一个有着较短证明的断言称为 NP 断言。在零知识证明研究中,只关心对 NP 语言类的证明/论证系统的一个重要原因是,绝大多数密码学的断言,如"我知道密文c对应的明文"等,都是 NP 断言(如果给出相应的密文和加密时所用的随机数即给出"证明",则很容易验证它)。

本小节将给出对所有 NP 语言的零知识证明系统。下面分成两步来达到这一目标。

(1) 构造一个对 NP 完全语言即汉密尔顿图的合理性错误为 1/2 的 3 轮零知识证明系

统。通过 Cook-Levin 定理，可以对任意 NP 语言构造出具有这种性质的 3 轮交互证明系统。

（2）通过顺序合成，在保持零知识的同时，把上述协议的合理性错误降低至$1/2^n$。这个合成后的协议将满足零知识证明系统的定义，具有零知识性和可忽略的合理性错误。

设 $G=(V,E)$ 为一 n 个顶点的无向图。$\boldsymbol{M}=(m_{i,j})_{1\leqslant i,j\leqslant n}$ 是它的邻接矩阵（Adjacent Matrix）：如果(i,j)为图 G 的边，即$(i,j)\in E$，则 $m_{i,j}=1$，否则 $m_{i,j}=0$。如果在图 G 中有一条封闭的路径 H 经过所有的顶点，则称 G 是汉密尔顿图，而把这条封闭的路径 H 称为汉密尔顿圈。判断一个图是否为汉密尔顿图是一个 NP 完全问题，所有的汉密尔顿图的集合就构成了一个 NP 完全的语言。

设 Com 为一个完美绑定且计算隐藏的非交互承诺方案（这种方案可以从单向置换来构造，具体实例可参见本书 12.1 节），$\boldsymbol{R}=(r_{i,j})_{1\leqslant i,j\leqslant n}$ 为一个随机矩阵。使用$\boldsymbol{R}$ 对邻接矩阵 $\boldsymbol{M}$ 中的值逐个进行承诺，记作 $C=\mathrm{Com}(M,R)=\{\mathrm{Com}(m_{i,j},r_{i,j})\}_{1\leqslant i,j\leqslant n}$，即 $\boldsymbol{C}$ 是一个由承诺值构成的矩阵。

Blum 提出的对图的汉密尔顿圈的零知识证明系统如下。

协议 5.1

公共输入：汉密尔顿图 G。

P 的私有输入：G 的汉密尔顿圈 H。

$P\rightarrow V$：随机选择一个置换 π 和随机矩阵$\boldsymbol{R}=(r_{i,j})_{1\leqslant i,j\leqslant n}$，令 $\boldsymbol{M}=(m_{i,j})_{1\leqslant i,j\leqslant n}$为图 $\pi(G)$ 的邻接矩阵。计算 $\boldsymbol{C}=\mathrm{Com}(\boldsymbol{M},\boldsymbol{R})$，并发送 $\boldsymbol{C}$。

$V\rightarrow P$：随机选择一个比特 e，并发送 e。

$P\rightarrow V$：如果 $e=0$，发送 $\pi,\boldsymbol{M}$ 和 $\boldsymbol{R}$（即打开所有的承诺）；如果 $e=1$，证明者将$\pi(H)$在 $\boldsymbol{C}$ 中对应位置的承诺值全部打开，即发送$(m_{i,j},r_{i,j})_{(i,j)\in\pi(H)}$。

V 的输出：如果 $e=0$，V 接受当且仅当所有的承诺值正确且 π 是 G 和$\boldsymbol{M}=(m_{i,j})_{1\leqslant i,j\leqslant n}$ 所确定的图之间的同构；如果 $e=1$，V 接受当且仅当所有打开的承诺值正确且其对应的位置形成一个 n 个顶点的圈。

现在来证明以下定理。

定理 5.1　如果存在单向置换，则对任何 NP 语言存在合理性错误为 1/2 的 3 轮零知识证明系统。

证明　只需证明协议 5.1 是合理性错误为 1/2 的 3 轮零知识证明系统即可。很容易验证完全性。

合理性。固定一个最优欺骗证明者策略 P'。注意到这里 P'具有无限计算能力，不失一般性，假定它为确定性图灵机。不难看出对于 P'，只有 3 种欺骗概率：0、1/2、1。如果 P'的欺骗概率大于 1/2，则它能以概率 1 使得诚实验证者 V 接受一个非汉密尔顿图G，即固定 P'的第一步消息，它既能正确回答诚实验证者 V 的挑战 $e=0$，又能回答诚实验证者的挑战 $e=1$。注意到以下两个事实。

（1）在这两次回答中，承诺方案 Com 的完美绑定性使得 P'在相同位置上的承诺被打开成同一个值。

（2）利用 P'在回答挑战$e=0$ 时得到的同构 π 和 P'在回答挑战$e=1$ 时得到的汉密尔顿圈，可以重构图 G 的汉密尔顿圈 H。

上述事实证明了 G 确实是一个汉密尔顿图,这与假设 G 是非汉密尔顿图矛盾。

零知识性。本质上,模拟器通过预先猜测验证者 V' 的挑战比特 e 来重构验证者的观察,由于猜对的概率很高,它将在较短的时间内猜对并停机输出。

模拟器构造如下。

模拟器 S

(1) 为 V' 选择随机带 r。

(2) 随机选择比特 b。

(3) 如果 $b=0$,随机选择一个置换 π 和随机矩阵 $\boldsymbol{R}=(r_{i,j})_{1\leqslant i,j\leqslant n}$,令 $\boldsymbol{M}=(m_{i,j})_{1\leqslant i,j\leqslant n}$ 为图 $\pi(G)$ 的邻接矩阵,计算 $\boldsymbol{C}=\text{Com}(\boldsymbol{M},\boldsymbol{R})$;如果 $b=1$,随机选择一个 n 个顶点的简单圈 H' 和随机矩阵 $\boldsymbol{R}=(r_{i,j})_{1\leqslant i,j\leqslant n}$,令 $\boldsymbol{M}=(m_{i,j})_{1\leqslant i,j\leqslant n}$ 为图 H' 的邻接矩阵,计算 $\boldsymbol{C}=\text{Com}(\boldsymbol{M},\boldsymbol{R})$,并发送 $\boldsymbol{C}$。

(4) 如果接收到 V' 的挑战比特 $e=b$,执行诚实证明者的策略回答这一挑战并输出 V' 的输出;否则返回第(1)步重新执行上述步骤。

注意到一旦模拟器 S 随机选择的比特 b 恰好与 V' 的挑战比特 e 一致,它便可以回答这一挑战,从而产生一个可接受的证明。由于猜测正确的概率为 1/2,所以模拟器的期望循环次数为

$$\sum_{n=1}^{\infty} n(1/2)^n = 2$$

在每一个循环中,易见 S 的运行时间是多项式的,故 S 可在期望多项式时间内停机并输出。

接下来将证明 S 的输出与真实交互中 V' 的输出是计算不可区分的。

用反证法证明这一事实:如果这两个分布是计算可区分的,则可构造算法打破承诺方案的计算隐藏性。这里将用到前面已经提到的一个简单事实:如果 S 重构的观察与真实交互中 V' 的观察计算不可区分,则 S 的输出与真实交互中 V' 的输出计算不可区分。

设图 G 的汉密尔顿圈为 H,π 为 H 与模拟器 S 在最后一个循环中选中的简单圈 H' 之间的同构,$H'=\pi(H)$。注意到当模拟器 S 正确地猜到 $b=e=0$ 时,它所重构的观察与真实交互中 V' 的观察服从同一分布,而当它正确地猜到 $b=e=1$ 时,它所重构的观察与真实交互中 V' 的观察仅在承诺矩阵 $\boldsymbol{C}$ 中某些位置上的被承诺值不同:在 S 所计算出来的 $\boldsymbol{C}$ 中,被承诺的邻接矩阵 $\boldsymbol{M}$ 中对应的 $\pi(G)$ 除了 H' 之外的边的位置上的值为 0,而在真实的交互中这些位置上的值为 1。这些不同值的位置共有 $|E|-n$ 个,这里 $|E|$ 为图 G 的总边数。

现在构造一系列混合(Hybrid)算法来证明 S 的输出与真实交互中 V' 的输出计算不可区分。

几点说明如下:

(1) 下面所有算法均为非一致算法,它们把图 G 的汉密尔顿圈 H 当成辅助输入。所谓非一致(Non-uniform)多项式时间图灵机是一个二元组 $(M,\bar{a})$,其中 M 为一个多项式时间图灵机,$\bar{a}=(a_1,a_2,\cdots)$ 为一个无限序列,$|a_n|=\text{poly}(n)$($\text{poly}(n)$ 是一个固定的多项式),$\bar{a}$ 也称为建议(Advice)序列。给定输入 x,图灵机 M 把 $a_{|x|}$ 当成额外的输入然后运行。注意到这里对于同样规模的问题 x,图灵机 M 所得到的"建议"是相同的。即对任意 x_i、x_j,$|x_i|=|x_j|$,M 的额外输入都是 $a_{|x_i|}=a_{|x_j|}$。为了方便,经常把某个图灵机 M 直接称为非一致图灵机而省略其"建议"序列。由于非一致图灵机能够得到额外的"建议",他的计算能力比普

通的图灵机稍强一些。非一致多项式时间图灵机等价于多项式规模的电路簇。可以类似地定义3个等价物,非一致概率多项式时间图灵机(及算法)和概率多项式规模的电路簇,这里将交替使用这3个概念。注意到能在概率多项式时间内识别的语言一定能在非一致多项式时间内识别,故非一致多项式时间图灵机(算法、电路)和非一致概率多项式时间的图灵机(算法、电路)在密码学意义上并没有区别,但有时为了清晰地描述某个具体的攻击算法,加上了"概率"一词,虽然在本章的定义中都省略了它。

(2) 预先随机选择一个随机置换 π,下面所有算法将使用这个固定的 π。这样做的好处是在分析中讨论的被承诺邻接矩阵是同一个矩阵 $\boldsymbol{M}$,方便比较那些具有不同被承诺值的位置。

(3) 把 $\pi(G)$ 除了 $H'=\pi(H)$ 之外的边按字典序排序,每条边对应一个有序自然数 i。

非一致模拟器 S^0:与模拟器 S 相同,除了第(3)步中在 $b=1$ 的情况下,S^0 使用简单圈 $H'=\pi(H)$,而不是随意选择 H'。

非一致模拟器 $S^i(1\leqslant i\leqslant |E|-n)$:与模拟器 S^{i-1} 相同,除了第(3)步中在 $b=1$ 的情况下,S^i 设置的邻接矩阵 $\boldsymbol{M}$ 中,$\pi(G)$ 除了 $H'=\pi(H)$ 之外第 i 条边对应的位置为1。

容易观察到以下几点事实。

(1) 尽管 S^i 设置的邻接矩阵可能不是 $\pi(G)$ 的邻接矩阵,但它在正确猜测到 $e=1$ 的情况下仍能响应这一挑战,应为那些相应位置上非正常的被承诺值并未被打开。

(2) 在图 G 是汉密尔顿图的前提下,随机选择简单圈 H' 与随机选择 π 并设置 $H'=\pi(H)$ 的效果相同。这使得事实上 S^0 的输出与 S 的输出服从同一分布。

(3) 非一致模拟器 $S^{|E|-n}$ 的输出与真实交互中 V' 的输出服从同一分布。

为简单起见,把 S^i 的输出仍记为 S^i,注意到它是一个由它自身的随机带所索引的随机变量。真实交换中 V' 的输出简记为 $\text{output}(V')$。

如果对于足够大的 n,存在概率多项式时间算法 D 和多项式 $p(n)$ 使得

$$|\Pr[D(S)=1]-\Pr[D(\text{output}(V'))=1]|>1/p(n)$$

则有

$$\begin{aligned}
&|\Pr[D(S)=1]-\Pr[D(\text{output}(V'))=1]|\\
=&|\Pr[D(S^0)=1]-\Pr[D(S^{|E|-n})=1]|\\
=&|\Pr[D(S^0)=1]-\Pr[D(S^1)=1]+\Pr[D(S^1)=1]\\
&-\cdots-\Pr[D(S^{|E|-n-1})=1]+\Pr[D(S^{|E|-n-1})=1]\\
&-\Pr[D(S^{|E|-n})=1]|>1/p(n)
\end{aligned}$$

进而存在某个 i 使得

$$|\Pr[D(S^{i-1})=1]-\Pr[D(S^i)=1]|>1/(|E|-n)p(n)$$

此不等式的左边可以写成

$$\begin{aligned}
&|\Pr[D(S^{i-1})=1]-\Pr[D(S^i)=1]|\\
=&|\Pr[D(S^{i-1})=1\mid b=0]\Pr[b=0]\\
&+\Pr[D(S^{i-1})=1\mid b=1]\Pr[b=1]\\
&-\Pr[D(S^i)=1\mid b=0]\Pr[b=0]\\
&-\Pr[D(S^i)=1\mid b=1]\Pr[b=1]|
\end{aligned}$$

这里 $b=1(b=0)$ 是最后一个循环中比特 b 的值。注意到 $\Pr[b=0]=\Pr[b=1]=1/2$ 并且

$\Pr[D(S^{i-1})=1|b=0]=\Pr[D(S^i)=1|b=0]$,则有

$$|\Pr[D(S^{i-1})=1|b=1]=\Pr[D(S^i)=1|b=1]|$$
$$>2/(|E|-n)p(n)$$

这表明算法 D 能够区分 S^{i-1} 的输出与 S^i 的输出,即能够区分 V' 在 S^{i-1} 最后一个循环中的观察与它在 S^i 最后一个循环中的观察。注意到在最后一个循环中,当 $b=1$ 时由 S^{i-1} 所重构的 V' 的观察与由 S^i 所重构的 V' 的观察唯一的区别在于,S^i 设置的邻接矩阵 $\boldsymbol{M}$ 中,$\pi(G)$ 除了 $H'=\pi(H)$ 之外第 i 条边对应的位置为 1,而 S^{i-1} 在相应的位置上设置的值为 0,此外,这个位置上的承诺在 S^{i-1} 或 S^i 发送给 V' 的最后一条消息中并未被打开,这使得容易构造一个非一致多项式时间的算法,打破该位置上承诺方案 Com 的计算隐藏性,从而导致了矛盾。

定理 5.2 如果存在单向置换,则顺序执行 n 次(证明者和验证者在每一次执行时独立选择新的随机带),协议 5.1 便得到了一个对汉密尔顿图的合理性错误为 $1/2^n$ 的零知识证明系统。

这个定理的证明留给读者。这里想强调顺序合成并不影响协议的零知识性:只需对每个子协议依次调用模拟器 S 即可。虽然模拟器的运行时间拉长了 n 倍,但它仍是期望多项式时间。

5.1.4 知识的零知识证明

大家知道,在成员的零知识证明中,证明者向验证者泄露了 1b 的信息,即 $x\in L$。但在某些应用场合,要求证明系统不能泄露任何信息。例如,在大多数现有的身份识别技术(如 ID 卡、信用卡、计算机口令、PIN 号等)中,证明者 P 通过提交记在心中或印在卡上的一个词 $i(P)$ 来证明他的身份。这样,一个与不诚实的验证者合作的敌手就能得到该卡的一个副本或者知道了这个词 $i(P)$,从而敌手可使用 $i(P)$ 来假扮 P 享受 $i(P)$ 所允许的访问或服务。解决这个问题的一个办法是证明者 P 使用零知识证明来使验证者 V 相信他知道 $i(P)$,而不泄露 $i(P)$ 的任何一个比特。这种零知识证明就是本节将要介绍的知识的零知识证明。

设 (P,V) 是一个交互协议,P 和 V 均具有多项式时间的计算能力,公共输入为 $x\in\{0,1\}^*$,P 拥有一条私有知识带 S。设 R 是 $\{0,1\}^*$ 上的一个多项式时间的二元关系(这个关系是公开知道的)。称 R 是 $\{0,1\}^*$ 上的一个二元关系,是指 R 是 $\{0,1\}^*\times\{0,1\}^*$ 的一个子集。如果 $(x,w)\in R$,称 x 和 w 满足关系 R,记为 $R(x,w)=$真,否则称 x 和 w 不满足关系 R,记为 $R(x,w)=$假。通常称 R 是 $\{0,1\}^*$ 上的一个多项式时间的二元关系,是指 R 是满足下列两个条件的 $\{0,1\}^*$ 上的一个二元关系:① $|w|$ 不超过 $|x|$ 的多项式;② 对任何 $x,w\in\{0,1\}^*$,可在 $|x|$ 的多项式时间内检测出是否 $R(x,w)=$真。例如,可以考虑下列的关系 $R(x,w)$:w 是 x 模一个素数 Q 的离散对数或 w 是 x 的一个完全分解。

我们说 (P,V) 对多项式时间关系 R 是一个知识的交互证明系统(Interactive Proof System of Knowledge),如果它满足下列两个条件:

(1) 完全性(Completeness):对所有的充分长的 x,如果在 P 的知识带 S 上存在一个 w 使得 $R(x,w)$ 为真,并且 P 和 V 都遵循协议,那么 V 将以很大的概率接收 x,即对每一个 $k>0$ 和充分长的 x:存在 $w\in S$,使 $R(x,w)=$真,V 至少以 $1-|x|^{-k}$ 的概率接收 x(即相信 P 知道使 $R(x,w)$ 为真的 w)。

(2) 合理性(Soundness)：对每一个 $k>0$，存在一个多项式时间的概率图灵机 M(M 被允许在没有修改或检查它的带子的情况下，可重置和重新运行 P'多项式次)，使得对每一个 $c>0$，对所有的 P'(可能不诚实)，P'的随机带 RP'和充分长的 x，如果将 x 作为(P',V)的公共输入，V 至少以 $|x|^{-k}$ 的概率接收 x，那么将 x 作为(P',M)的公共输入，(P',M)至少以 $1-|x|^{-c}$的概率输出一个 w'使得 $R(x,w')$为真。

条件(1)是说，如果 P 知道 w，V 将以很大的概率接收 P 对 x 的证明。条件(2)是说，如果 P 不知道 w，V 将以很小的概率接收 P 对 x 的证明。所说的 P 知道 w，是指存在某一多项式时间概率图灵机 M(M 被允许在没有修改或检查它的带子的情况下，可重置和重新运行 P 多项式次)，使得 M 和 P 交互的结果是 w。将上述定义中的 M 称为知识抽取器。知识的证明系统的概念有多种定义，这些定义之间有一些微小的差别，有的书中也将上述定义的知识证明系统称为强知识的证明系统。

我们说一个协议(P,V)对多项式时间关系 R 是完美(统计、计算)零知识的，如果对任何多项式时间的概率图灵机 V'(V'带有一条附加输入带，V'的附加输入记为 H)，随机变量族 $\text{View}_{P,V'}=\{\text{View}_{P,V'}(x,H)\}_{(x,H)\in L'}$关于语言 $L'=\{(x,H)|$存在$w\in S$，使 $R(x,w)=$真，$|H|$不超过$|x|$的多项式$|\}$是完美(统计、计算)可逼近的。我们说(P,V)对多项式时间关系 R 是一个知识的完美(统计、计算)零知识证明系统，如果它对 R 是一个知识的交互证明系统并且是完美(统计、计算)零知识的。

由定义知，知识的完美零知识证明系统一定是知识的统计零知识证明系统，知识的统计零知识证明系统一定是知识的计算零知识证明系统。通常将知识的计算零知识证明系统称为知识的零知识证明系统，简称知识的零知识证明。

文献[14]中讨论了一种与 FFS 模型相类似的模型，在这种模型中，证明者 P 能证明 $I\in L$ 或 $I\notin L$，验证者 V 知道 $I\in L$ 还是 $I\notin L$，并相信 P 的证明，除此之外，V 没有从 P 那里获得任何附加信息。但被动的窃听者 C(不允许参加或插入协议)不能确定 $x\in L$ 还是 $x\notin L$，并且不能确定 P 的证明是否可信。关于知识的零知识证明的定义已给出了许多种[6,15]，本小节介绍的是 FFS 的定义[6]，文献[6]中也说明了知识的零知识证明系统的存在性，它指出：在 NP∩Co_NP 中的任何问题都有一个知识的零知识证明系统。不过由 FFS 的定义易知文献[12]中关于所有 NP 问题的成员的交互零知识证明也是知识的交互零知识证明。关于知识的零知识证明也可参阅文献[16]、[17]。知识的零知识证明在安全协议的研究中有着重要的作用，作为知识的零知识证明的应用，在这里暂举一例。

下面来描述一个身份识别协议，这个协议的目的是能使证明者 P 向验证者 V 证明他的身份，而事后 V 又不能冒充 P。这里假定存在一个可信任的中心，该中心的唯一目的是公开两个形式为 $4r+3$ 的大素数的乘积 n，这种整数称为 Blum 整数，在各种密码应用中使用这种整数作为模，Blum 整数的最有用的特性之一是：-1 是模 n 的一个非二次剩余，并且 -1 的 Jawbi 符号为 $+1$，即 $\left(\frac{-1}{n}\right)=1$。在公开 n 后，中心可以被取消或关闭，因为它再没有其他作用。协议中的每个人都能使用 n，但没有人能知道 n 的分解。

P 的秘密身份证 $i(P)$的产生过程如下。

(1) 在 Z_n 中随机选择 k 个数 $S_1,S_2,\cdots,S_k$。

(2) 随机地，独立地选择 $I_j=\pm\frac{1}{S_j^2} \bmod n, 1\leqslant j\leqslant k$。

(3) 公开 $I_1, I_2, \cdots, I_k$，将 $P_i(P)=(I_1, I_2, \cdots, I_k)$ 作为 P 的公开身份证；保密 $S_1, S_2, \cdots, S_k$，将 $i(P)=(S_1, S_2, \cdots, S_k)$ 作为 P 的秘密身份证。

验证者 V 知道公开的 n 和 $P_i(P)$，证明者 P 想使 V 相信他知道 $i(P)$，但又不想对 V 泄露任何信息。为了达到这一点，P 和 V 重复执行下列步骤 t 次(轮数 t 能降低 P 的欺骗概率)。

(1) P 选择一个随机数 R，计算 $X=\pm R^2 \bmod n$，并将它们之中的一个发送给 V。

(2) V 随机选择一个向量 $(E_1, \cdots, E_k) \in Z_2^k$，并发送给 P。

(3) P 计算 $Y = R \prod_{\substack{E_j=1 \\ 1\leqslant j\leqslant k}} S_j \pmod n$，并将 Y 发送给 V。

(4) V 验证是否 $X = \pm Y^2 \prod_{\substack{E_j=1 \\ 1\leqslant j\leqslant n}} I_j \pmod n$。

在上述的身份识别过程中，P 没有提交他的秘密身份证，也没有去向 V 证明他的公开身份证的合法性，而是向 V 通过显示他拥有关于他的秘密身份证的知识来证明他的公开身份证的合法性。

文献[6]中证明了在假定求模 n 的平方根困难的条件下，上述协议是 $\{S_j\}_{j=1}^{k}$ 的知识的零知识证明，其中 $k=O(\log(\log n))$，$t=O(\log n)$，n 的因子分解是未知的。

在前面的论述中，将协议中的参加者(如证明者、验证者、欺骗者等)都抽象成一台概率图灵机。一台机器是由它所运行的程序或算法来控制的，程序或算法和机器之间是相互决定的，由此可见，可以将机器视作程序或算法，在现实世界中，有时也可以将机器视作人、现实的计算机等实体。

5.1.5 知识的证明/论证系统

零知识协议的可合成性(Composibility)研究最开始主要是为了降低合理性的错误概率，如 5.1.3 小节中提到的顺序合成(Sequential Composition)和并行合成(Parallel Composition)，但后来研究者们把目光转向了研究如何在更具敌意的环境中来保持系统的零知识性，由此引出了更高级的合成——并发合成(Concurrent Composition)。在本节中，主要考虑前两种简单的情形。

顺序合成是指一个协议自身被多次执行，每次执行都发生在上次调用结束后并且诚实的一方在一次执行完成后将独立选取新的随机数进入下一次执行。顺序合成已被证明是保持零知识性的，并且合理性错误概率随着执行的次数增加呈指数级下降。如同定理 5.2 所述，如果系统 (P,V) 具有零知识性并且合理性错误概率为 1/2，那么对它进行 n 次顺序执行所形成的新系统 (P',V') 仍保持零知识性且合理性错误概率下降为 $1/2^n$。

注意到以这样的方式来降低合理性错误概率极大地牺牲了效率，以轮复杂性(在协议执行中每发送一个消息称为一轮)为例，如果原来的协议 (P,V) 轮数为 c，则新协议 (P',V') 是一个 cn 轮的协议。

并行合成是指一个协议的多个同步执行，这里同步指的是不同的执行中的证明者和验证者按照相同的步调发送消息，即验证者同时发送第 i 条消息，接着所有的证明者向对应的验证者发送第 $i+1$ 条消息。最后验证者接受当且仅当所有的这些执行产生的副本都是可接受的。

并行合成最大的好处就是在保持原来协议的轮复杂性同时降低了合理性错误，在本节中将看到这一点。但它也带来了一个巨大的问题：在一般情况下，无法证明合成后的协议仍具有零知识性。

虽然并行合成在某种程度上会损害零知识性，但它保持一种比零知识性稍弱但仍具有极强应用价值的隐私性：证据不可区分。简单地讲，证据不可区分的证明/论证系统具有这样的性质：对于一个断言的两个证据，在执行一个证据不可区分的证明系统后，验证者不能判定证明者在这个证明中使用的是哪个证据。

设(P,V)为语言 L 的一个交互证明/论证系统。如果对于任意的多项式规模的电路 C，公共输入 $x\in L$ 和它的两个证据 w_0、w_1(使得$(x,w_0)\in R_L$ 和$(x,w_1)\in R_L$)，下面两个随机变量族：

(1) $\{(P(w_0),C)(x)\}_{x\in L}$。

(2) $\{(P(w_1),C)(x)\}_{x\in L}$。

是计算(完美、统计)不可区分的，则称证明/论证系统(P,V)是计算(完美、统计)证据不可区分的。

显然，一个协议是零知识的，它必然是证据不可区分的，但反之则不成立。尽管证据不可区分是一个稍弱的概念，读者将在以后的讨论中会看到它的威力。事实上，证据不可区分证明/论证系统是构造几乎所有具有高安全性零知识协议的基本模块。

现在来构造一个具有很低合理性错误的证据不可区分证明系统。它是协议5.1 的一个简单的并行版本。

协议 5.2(协议 5.1 并行版)

公共输入：汉密尔顿图 G。

P 的私有输入：G 的汉密尔顿圈 H。

$P\to V$：随机选择 k 个置换 $\pi_1,\cdots,\pi_k$ 和 k 个随机矩阵 $\boldsymbol{R}_1=(r_{i,j}^1)_{1\leqslant i,j\leqslant n},\cdots,\boldsymbol{R}_k=(r_{i,j}^k)_{1\leqslant i,j\leqslant n}$，令 $\boldsymbol{M}_1=(m_{i,j}^1)_{1\leqslant i,j\leqslant n},\cdots,\boldsymbol{M}_k=(m_{i,j}^k)_{1\leqslant i,j\leqslant n}$分别为图$\pi_1(G),\cdots,\pi_k(G)$的邻接矩阵。计算 $\boldsymbol{C}_1=\mathrm{Com}(M_1,R_1),\cdots,\boldsymbol{C}_k=\mathrm{Com}(M_k,R_k)$，并发送$\boldsymbol{C}_1,\cdots,\boldsymbol{C}_k$。

$V\to P$：随机选择比特串 $e=(e_1,\cdots,e_k)$，并发送 e。

$P\to V$：对于所有的 l，$1\leqslant l\leqslant k$，如果 $e_l=0$，发送 π_l、M_l 和 R_l；如果 $e_l=1$，证明者将$\pi_l(H)$在 C_l 中对应位置的承诺值全部打开，即发送$(m_{i,j}^l,r_{i,j}^l)_{(i,j)\in\pi_l(H)}$。

V 的输出：对于所有的 l，$1\leqslant l\leqslant k$，V 用在协议 5.1 单次执行时同样的方式检查第 l 个副本是否正确，它接受当且仅当它接受全部 k 个副本。

读者可能会想到利用 5.1.3 小节中构造的模拟器来证明这个并行版本的零知识性。但这种经典的重置模拟方式并不可行。对于协议 5.1，模拟器猜测验证者挑战比特的概率很高，所以它能在很短时间内猜测正确；对于协议 5.2，当 k 值比较大时，它猜测正确的概率可忽略，所以不可能在期望多项式时间内完成模拟。

虽然协议 5.2 不是零知识的，但它是一个证据不可区分证明系统。

定理 5.3　如果存在单向置换，协议 5.2 是一个合理性错误为 $1/2^k$ 的计算证据不可区分证明系统。换句话说，如果存在单向置换，则对任何 NP 语言存在证据不可区分证明系统。

证明　完全性显然。这里不证明协议 5.2 的合理性错误，而是在本小节后半部来证明

它的一个比合理性更强的性质。

证据不可区分性。设图 G 具有两个不同的汉密尔顿圈 H_0 和 H_1。进一步假设存在多项式时间的验证者 V'能够区分证明者在一个协议的执行中使用的证据是H_0 还是 H_1，即对于足够大的 n，存在概率多项式时间算法 D 和多项式 $p(n)$使得

$$|\Pr[D((P(H_0),V')(x))=1]-\Pr[D((P(H_1),V')(x))=1]|>1/p(n)$$

利用混合论证的办法证明它不可行，否则将打破协议 5.1 的零知识性。考虑以下证明者策略。

证明者策略 P^0：与证明者 $P(H_0)$完全一致，即在协议 5.1 的所有 k 次执行中，证明者 P^0 使用同一个证据 H_0。

证明者策略 $P^l(1\leqslant l\leqslant k)$：除了在第 l 次执行子协议 5.1 时使用证据 H_1，其余与证明者 P^{l-1}相同

根据假设，有

$$\begin{aligned}&|\Pr[D((P(H_0),V')(G))=1]-\Pr[D((P(H_1),V')(G))=1]|\\=&|\Pr[D((P^0,V')(G))=1]-\Pr[D((P^k,V')(G))=1]|\\=&|\Pr[D((P^0,V')(G))=1]-\Pr[D((P^1,V')(G))=1]\\&+\Pr[D((P^1,V')(G))=1]-\cdots\Pr[D((P^{k-1},V')(G))=1]\\&+\Pr[D((P^{k-1},V')(G))=1]-\Pr[D((P^k,V')(G))=1]|>1/p(n)\end{aligned}$$

进而存在某个 i 使得

$$|\Pr[D((P^{l-1},V')(G))=1]-\Pr[D((P^l,V')(G))=1]|>1/kp(n)$$

注意到 P^l 与证明者 P^{l-1}唯一的区别在于它第 l 次执行子协议 5.1 时使用证据 H_1，同时注意到子协议 5.1 的计算零知识性和证明者在执行子协议时使用独立的随机带。考虑以下混合分布。

随机变量 HS：HS 是有关 V'的观察的一个随机变量。在这个观察中，调用模拟器 S 来输出第 l 次执行子协议 5.1 时的观察，在所有其他的协议执行中，使用证明者策略 P^{l-1}与 V'交互。

由于协议 5.1 的零知识性，对于任意的多项式 $q(n)$，有

$$|\Pr[D((P^{l-1},V')(G))=1]-\Pr[D(HS)=1]|<1/q(n)$$

与

$$|\Pr[D((P^l,V')(G))=1]-\Pr[D(HS)=1]|<1/q(n)$$

进而对于任意的多项式 $q(n)$，有

$$|\Pr[D((P^{l-1},V')(G))=1]-\Pr[D((P^l,V')(G))=1]|<1/q(n)$$

这与前面的不等式矛盾，因此协议 5.2 是计算不可区分的。

接下来讨论关于合理性的一种加强的变体：知识的证明/论证。本质上，合理性要求当 NP 断言错误时，敌意的证明者无法使得诚实的验证者接受这一断言；而知识的证明/论证则进一步要求一旦诚实的验证者接受某个断言，则能高效地提取出这个 NP 断言的一个证据。

设(P,V)为语言 L 的一个合理性错误为 $e(n)$的证明/论证系统。对于所有的x 和所有的多项式规模的电路 C，如果 $\Pr[(C,V)(x)=1]=p(n)>e(n)$，则存在一个概率多项式时间的知识提取器 E，给定 C 的代码或把 C 当成 Oracle 来调用，它能以 $p(n)-e(n)-\mu(n)$的

概率输出 w 使得$(x,w)\in R_L$，这里 $\mu(n)$为可忽略函数，那么称(P,V)为语言 L 的一个知识的证明(论证)系统。

不难验证协议 5.1 是合理性错误为 $1/2^k$ 的知识的证明系统。对于某个图 G，观察到如果某个证明者 P'能以大于 $1/2^k$ 的概率，使得诚实验证者接受 G 为汉密尔顿图这一断言，则它至少能回答诚实验证这两个不同的挑战比特串，不妨设为 $e^0=(e_1^0,\cdots,e_k^0)$和 $e^1=(e_1^1,\cdots,e_k^1)$，其中 $e_l^0\neq e_l^1$。这就说明 P'在执行第 l 个子协议 5.1 时既能回答验证者的挑战比特 $e=0$ 又能回答 $e=1$，这样就能够从这两个回答中计算出图 G 的汉密尔顿圈。读者可以检验这一提取汉密尔顿圈的成功概率符合定义要求。

5.2 非交互零知识证明理论

现在考虑这样一种情况：P 和 V 是两个数学家，P 动身作一次长的环球旅行，在旅行期间他继续从事他的数学研究。我们希望 P 每发现一个新定理的证明都能通过给 V 写一张明信片用零知识证明他的论断的合法性。但是因为 P 没有固定或可预测的地址并且在任何邮件寄到 P 寄来的明信片上写的地址之前他已离开，所以即使 V 乐意与 P 交互，也无法与 P 取得联系，可见，前一节介绍的交互零知识证明已不再适合这种应用环境，这里需要一个单项交互，即只是从 P 到 V。适用于这种应用环境的零知识证明就是下面将要介绍的非交互零知识证明(Non-Interactive Zero-Knowledge Proof)。像交互零知识证明的研究一样，目前，在非交互零知识证明的研究中，人们也主要关心两种模型：一种是成员或定理的非交互零知识证明系统；另一种是知识的非交互零知识证明系统。本节只讨论成员或定理的非交互零知识证明系统，简称非交互零知识证明。对知识的非交互零知识证明系统，这里不作讨论，感兴趣的读者可参阅文献[18]、[19]。需要指出的是，知识的非交互零知识证明系统的构造不像知识的交互零知识证明系统的构造那样容易。

首先应该认识到去掉交互也是有代价的，在证明中需要引入一个可信的第三方。读者容易验证对于非平凡语言，在没有交互和可信第三方的情况下，则无法构造一个零知识证明系统。在现实世界中，为了提高效率或达到其他目的，在安全系统中通常假定了可信第三方，如 PKI。这就使得非交互零知识协议有着非常广阔的应用前景。一方面，它不需要交互；另一方面，在现实中卷入可信第三方是可行的。

这个非交互模型有两个必要成分，即公共参考串和计算困难性。公共参考串是由可信第三方提供的，证明者和验证者都能够读取公共参考串。这种证明非常简单，它就是证明者发送给验证者的一条消息，然后验证者根据这条消息去决定是否接受。

具体地讲，一个非交互证明系统由一个公共参考串生成算法 K(由可信第三方来运行)、一个证明者 P 和一个概率多项式时间验证者 V 构成，这里诚实证明者有时是指数时间的。如果证明者能被概率多项式时间实现，则称相应的证明系统为高效证明者(Efficient Prover)证明系统。给定安全参数 n，公共参考串生成算法生成一个多项式长度的公共参考串 σ。证明者 P 把公共输入 x、相应的证据 w 及公共参考串σ 当成输入，生成一个证明 π。验证者 V 验证三元组(x,σ,π)并输出接受或拒绝。

5.2.1 非交互证明系统

与交互证明系统相对应，在介绍非交互零知识证明之前首先来描述非交互证明系统。

在一个非交互证明系统中也有两个分别称为证明者和验证者的参加者。证明者知道某一定理的证明，他希望向验证者证明他的确能证明这一定理。一个语言L的非交互证明系统由两个阶段构成：第一个阶段是预处理阶段，主要建立证明者和验证者拥有的某些共同信息以及他们各自拥有的某些秘密信息，这个预处理阶段独立于定理证明阶段，而且允许证明者和验证者之间进行交互；第二个阶段是定理证明阶段，证明者选择并向验证者证明定理，这个定理证明阶段是非交互的。证明者和验证者可分别考虑作$P=(P_1,P_2)$和$V=(V_1,V_2)$。P_1和P_2是任意概率图灵机（一般认为P_2具有无限的计算能力），V_1和V_2都是概率多项式时间图灵机（也称多项式时间的概率图灵机）。设k是一个安全参数。

我们说$P=(P_1,P_2)$和$V=(V_1,V_2)$对一个语言$L\subset\{0.1\}^*$构成一个非交互证明系统，如果对所有充分大的k，下列两个阶段的要求都满足：

(1) 预处理阶段。通过P_1和V_1的交互产生3个输出σ、S_P、S_V，其中σ是P_1和V_1共同拥有的公共信息（亦称参考串），S_P是P_1的秘密信息，S_V是V_1的秘密信息。如果P_1不采纳规定的协议，那么V_1以很高的概率输出“拒绝”。

(2) 定理证明阶段（所谓“定理”意指手边的一个串$x\in L$）。P_2非交互地向V_2证明定理（关于任何输入的通信仅是从P_2到V_2的一个消息）。

① 完全性。对每一个$x\in L\cap\{0.1\}^*$，$P(V_2(1^k,\sigma,S_V,x,\beta)=$接收：$(\sigma,S_P,S_V)\leftarrow(P_1\leftrightarrow V_1)_{(1^k)};\beta\leftarrow P_2(1^k,\sigma,S_P,x))\geqslant 1-2^{-2k}$，即$V_2$以很高的概率接收$P_2$的证明。

② 合理性。对每一个$x\in\bar{L}\cap\{0.1\}^k$及每一个概率图灵机P_2'，$P(V_2(1^k,\sigma,S_V,x,\beta)=$接收：$(\sigma,S_P,S_V)\leftarrow(P_1\leftrightarrow V_1)_{(1^k)};\beta\leftarrow P_2'(1^k,\sigma,S_P,x))\leqslant 2^{-2k}$，即$V_2$以很小的概率接收$P_2$的证明。

其中，$\bar{L}$表示L的补，即不属于L中的元素所组成的集合。1^k表示k个1的级联。$(P_1\leftrightarrow V_1)_{(x)}$表示关于输入$x$的输出的概率空间，在这些输出中，也许一些是$P_1$或$V_1$的秘密信息，而另一些是它们的共同信息。如果$S$是一个概率空间，那么$x\leftarrow S$表示根据$S$的概率分布随机选择一个元素分配给$x$的一个算法。如果$S$是一个有限集合，那么$x\leftarrow S$表示$S$中均匀地随机选择一个元素分配给$x$的一个算法。$P(p(x,y,\cdots):x\leftarrow s,y\leftarrow T,\cdots)$表示在依次执行完算法$x\leftarrow S,y\leftarrow T$等后，关系$p(x,y,\cdots)$是真的概率。

注：①上述定义是一个更一般的定义，预处理阶段获得的σ、S_P、S_V中的某些可以是空串；②在某些情况下，也允许非交互证明系统把以前的定理及其证明的历史作为一个附加输入。

下面将对一类特殊语言L描述一个非交互证明系统，在此之前，先介绍一点预备知识。

关于合数的测试有以下结果[20]：存在一个多项式时间的概率算法TEST(·,·)使得：①如果x是合数，那么在集合$A=\{r|r$是一个串并且r的长度为$|x|\}$中，至少有$\frac{3}{8}$使得TEST$(x,r)=$合数；②如果x是素数，那么对所有的r，都有TEST$(x,r)=$素数。

整数x的长度是指x的二进制表示的串的长度，表示为$|x|$。要特别注意在不同的场合，$|x|$的含义不同。如果x是一个实数，$|x|$表示x的绝对值。如果x是一个串，$|x|$表示x的长度。如果x是一个集合，$|x|$表示x中元素的个数。

对正整数x，用J_x^{+1}和J_x^{-1}分别表示Z_x^*中Jacobi符号为+1和−1的集合。

我们说一个正整数x是正规的，如果$|J_x^{+1}|=|J_x^{-1}|$。用Regular(s)表示具有s个不同

素因子的全体正规整数之集。关于正规整数，由 Jacobi 符号的定义易知下列结论成立：一个奇整数 $x\in \text{Regular}(s)$，当且仅当它有 s 个不同的素因子并且不是一个完全平方数。

设奇整数 $x\in \text{Regular}(s)$，定义等价关系 $\sim_x$ 为 $y_1\sim_x y_2$，当且仅当 y_1y_2 是模 x 的一个二次剩余。则 Z_x^* 由等价关系 $\sim_x$ 分成 2^s 个等价类且所有的等价类含有的元素个数都相同。特别地，J_x^{+1} 由等价关系 $\sim_x$ 分成 2^{s-1} 个等价类且所有的等价类含有的元素个数都相同，都等于模 x 的二次剩余的个数。

对 $n\in N$，设 $\text{SQNR}(n)=\{(x,y)\mid x\in \text{Regular}(2),x$ 为奇整数，$|x|\leqslant n,y\in J_x^{+1}$ 但 y 不是模 x 的一个二次剩余$\}$，$\text{SQNR}=\bigcup_{n\geqslant 1}\text{SQNR}(n)$。这里 $|x|$ 表示整数 x 的长度。设 $(x,y)\in \text{SQNR},z\in J_x^{+1}$，说 $s\in Z_x^*$ 是 z 的一个 (x,y) 根，如果 $z=s^2 \bmod x$ 或 $zy=s^2 \bmod x$。简记为 $s=\sqrt[(x,y)]{z}$。

下面就来描述一个对语言 SNQR 的非交互证明系统。

(1) 预处理阶段。P_1 和 V_1 共同拥有一个 n^3b 随机串 σ(即参考串)，设 $\sigma=\sigma_1\sigma_2\cdots\sigma_{n^2}$，每个 $\sigma_i(1\leqslant i\leqslant n^2)$ 的长度为 n，S_P 和 S_V 都是空串。

(2) 定理证明阶段。P_2 非交互地向 V_2 证明定理。P_2 和 V_2 的共同输入为 $(x,y)\in \text{SQNR}(n)$，P_2 和 V_2 的执行规则分别如下。

P_2 的执行规则为：对 $i=1,2,\cdots,n^2$，如果 $\sigma_i\in J_x^{+1}$，那么随机地选择并发送 $s_i=\sqrt[(x,y)]{\sigma_i}$。

V_2 的执行规则如下。

V_2.1. 验证 x 和 y，如果 x 是奇正整数并且 $y\in J_x^{+1}$，就继续执行 V_2.2，否则就停机并拒绝接收。

V_2.2. 验证 x 是不是一个完全平方数，如果不是，就继续执行 V_2.3，否则就停机并拒绝接收。

V_2.3. 验证 x 是不是一个素数幂，如果不是，就继续执行 V_2.4，否则就停机并拒绝接收。

V_2.4. 对每一个 $\sigma_i\in J_x^{+1}$，如果 $s_i=\sqrt[(x,y)]{\sigma_i}$，$V_2$ 就接收 P_2 的证明，否则就停机并拒绝接收。

因为 Jacobi 符号能在多项式时间内计算出，所以 V_2 的执行步骤 V_2.1 和 V_2.4 可在多项式时间内验证。V_2 的执行步骤 V_2.2 显然可在多项式时间内验证。而 V_2.3 可由下列步骤来完成：

V_2.3.1. 计算整数 α 和 w 使得 $x=w^\alpha$。如果存在整数 α 和 w 使得 $x=w^\alpha$，那么 α 的取值范围只能是 $1,2,\cdots,|x|$，通过搜索能找到整数 α 和 w 使得 $x=w^\alpha$。

V_2.3.2. 如果对所有的 $1\leqslant i\leqslant n^2$，$\text{TEST}(w,\sigma_i)=$素数，那么 V_2 停机并拒绝接收。

由执行步骤 V_2.3.1 和 V_2.3.2 以及合数的测试算法 $\text{TEST}(\cdot,\cdot)$ 可知，V_2.3 可在多项式时间内完成。综上所述，V_2 是多项式时间的。

现在来证明上述系统满足完全性和合理性。

完全性：如果 $(x,y)\in \text{SQNR}(n)$，那么 V_2.1 和 V_2.2 显然以概率 1 通过验证。下面着重讨论 V_2.3 和 V_2.4 通过验证的概率。

对任何固定的整数 $\bar{x}\in \text{Regular}(2)$，合数的测试算法 $\text{TEST}(\cdot,\cdot)$ 关于 σ_i 输出素数的概率至多是 $\frac{5}{8}$，这样 V_2.3 不能通过验证的概率至多是 $\left(\frac{5}{8}\right)^{n^2}$。因为对某一个 w，至多有

2^n 个 x 使得$(x,w)\in \mathrm{SQNR}(n)$，所以 $V_2.3$ 不能通过验证的概率至多是$2^n\times\left(\frac{5}{8}\right)^{n^2}$。当 n 充分大时，$2^n\times\left(\frac{5}{8}\right)^{n^2}\leqslant 2^{-2n}$。故当 n 充分大时，$V_2.3$ 通过验证的概率不小于 $1-2^{-2n}$。

当 $x\in \mathrm{Re\ gular}(2)$时，由前面的讨论知，在 J_x^{+1} 中有两个等价类，也就是 σ_i 或者是模 x 的一个二次剩余或者是与 y 在一个等价类中，在后者这种情况下，$y\sigma_i$ 是模x 的一个二次剩余，因此，$V_2.4$ 以概率 1 通过验证。

由上述讨论知，对充分大的 n，当$(x,y)\in \mathrm{SQNR}(n)$时，$V_2$ 以不小于 $1-2^{-2n}$的概率接收 P_2 的证明，说明此系统满足完全性。

合理性：如果$(x,y)\notin \mathrm{SQNR}(|x|\leqslant n,|y|\leqslant n)$，那么当 x 不是奇正整数或$y\notin J_x^{+1}$ 时，显然，$V_2.1$ 通不过验证的概率为 1。当 x 是奇正整数并且 $y\in J_x^{+1}$ 时，有下述两种情况：①$x\in \mathrm{Re\ gular}(2)$，但 y 是模 x 的一个二次剩余；②$x\notin \mathrm{Re\ gular}(2)$。在情况①下，对任何固定的输入$(\bar{x},\bar{y})$，$V_2.4$ 能通过验证的充要条件是所有的$\sigma_i(1\leqslant i\leqslant n^2)$都是模 $\bar{x}$ 的二次剩余，而 $J_{\bar{x}}$ 由等价关系$\sim_x$ 分成含相同个数的 $2^2=4$ 个等价类，所以每个 σ_i 是模$\bar{x}$ 的一个二次剩余的概率至多是$\frac{1}{4}$，这样，$V_2.4$ 能通过验证的概率至多是$\left(\frac{1}{4}\right)^{n^2}=2^{-2n^2}$。因为对某一固定的 $\bar{y}$，至多有 2^n 个$\bar{x}(|\bar{x}|\leqslant n)$使得$(\bar{x},\bar{y})\notin \mathrm{SQNR}$且 $\bar{y}$ 是模$\bar{x}$ 的一个二次剩余，所以 $V_2.4$ 能通过验证的至多是 $2^n\times 2^{-2n^2}=2^{-2n^2+n}$。当 n 充分大时，$2^{-2n^2+n}\leqslant 2^{-2n}$，因此，在情况①下，$V_2.4$ 能通过验证的概率不大于 2^{-2n}。在情况②下，或者(A)x 不是正规整数，或者(B)$x\in \mathrm{Re\ gular}(1)$，或者(C)$x\in \mathrm{Re\ gular}(s)$，$s\geqslant 3$。在情况(A)下，一个奇正整数 x 一定是一个完全平方数，这将在 $V_2.2$ 中被检测出。在情况(B)下，x 是一个素数幂，将在 $V_2.3$ 中被检测出。在情况(C)下，对任何固定的$(\bar{x},\bar{y})$，$\bar{x}\in \mathrm{Re\ gular}(s)$，$s\geqslant 3$，$V_2.4$ 能通过验证的充要条件是对每一个 $\sigma_i\in J_x^{+1}$，或者 σ_i 或者$\sigma_i\bar{y}$ 是模 $\bar{x}$ 的一个二次剩余，而 J_x^{+1} 至少有 4 个等价类，因而 σ_i 或$\sigma_i\bar{y}$ 是模 $\bar{x}$ 的一个二次剩余的概率至多是$\frac{1}{2}$。这样，$V_2.4$ 能通过验证的概率至多是$\left(\frac{1}{2}\right)^{n^2}=2^{-n^2}$。因为至多有 2^{2n}对$(x,y)\notin \mathrm{SQNR}(|x|\leqslant n,|y|\leqslant n)$，所以对任何输入 $V_2.4$ 能通过验证的概率至多是 $2^{2n}\times 2^{-n^2}=2^{-n^2+2n}$，当 n 充分大时，$2^{-n^2+2n}\leqslant 2^{-2n}$。故在情况②下，$V_2.4$ 能通过验证的概率不大于 2^{-2n}。综上所述，此系统满足合理性。

5.2.2 非交互零知识证明

非交互零知识证明是一种特殊的非交互证明系统，它要求在证明中不允许泄露任何有用的消息。像交互零知识证明系统一样，非交互零知识证明系统的证明者能使验证者相信 x 具有某种具体的特性(诸如 $x\in L$)，但执行完协议后，验证者自己仍然一点也不知道如何来证明 x 具有这个特性。

为了定义零知识性，先介绍一些记号。

对任何图灵机 V_1'，设$(P_1,V_1')(1^k)$表示由 P_1 和V_1'交互的输出、交互的历史和V_1'的掷硬币所构成的概率空间，把交互的历史和 V_1'的掷硬币合成一个元素 h。用(σ,S_P,S_V,h)表示这个空间的一个元素。

设 $P=(P_1,P_2)$，$V=(V_1,V_2)$是关于语言 L 的一个非交互证明系统。对每一对概率多

项式时间图灵机 $V'=(V_1',V_2')$(允许它们有附加的输入带)和 $x\in L\cap\{0,1\}^k$,将 V' 对 P 关于输入 x 的观察记为 $\text{View}_{V'}^{(P)}(1^k,x)$,这里

$\text{View}_{V'}^{(P)}(1^k,x)=\{(\sigma,S_V,h,\text{proof}):(\sigma,S_P,S_V,h)\leftarrow(P_1,V'_1)(1^k);\text{proof}\leftarrow P_2(1^k,\sigma,S_P,x)\}$是一个随机变量。

我们说一个关于语言 L 的非交互证明系统 $P=(P_1,P_2)$,$V=(V_1,V_2)$是完美(统计、计算)零知识的,如果对每一对概率多项式时间图灵机 $V'=(V_1',V_2')$,存在一个概率多项式时间图灵机 S 使得对每一个充分大的 k 及每一个 $x\in L\cap\{0,1\}^k$,$\text{View}_{V'}^{(P)}(1^k,x)$和 $S(1^k,x)$是完美(统计、计算)不可区分的。

注:①允许非交互零知识证明系统把以前的定理及其证明历史、交互历史作为一个附加输入,这里为了简单起见,没有考虑;②由定义易知,非交互完美零知识证明系统一定是非交互统计零知识证明系统,非交互统计零知识证明系统一定是非交互计算零知识证明系统;③如果证明者多次使用同一个参考串 σ 使验证者相信许多定理的正确性,那么产生的非交互证明系统也许不再是零知识的,这种性质称为有界性,具有这种性质的非交互零知识证明系统称为有界的非交互零知识证明系统,5.2.1小节中的例子就是一个有界的非交互完美零知识证明系统,下面将证明其零知识性。

为了说明5.2.1小节中的非交互证明系统是(完美)零知识的,首先构造一个概率多项式时间图灵机(模拟器)S,其运行程序如下。

输入:$(x,y)\in \text{SQNR}(n)$。

S.1.设 Proof=空串。

S.2.对 $i=1,2,\cdots,n^2$,随机地选择一个 n 比特整数 s_i,如果 $s_i\notin J_x^{+1}$,那么设 $\sigma_i=s_i$,否则掷一个硬币,如果是0,设 $\sigma_i=s_i^2 \bmod x$ 并把 s_i 级联到 Proof,如果是1,设 $\sigma_i=y^{-1}s_i^2 \bmod x$ 并把 s_i 级联到 Proof。

S.3.设 $\sigma=\sigma_1\sigma_2\cdots\sigma_{n^2}$。

输出:(σ,proof)。

因为在这个非交互证明系统中,S_P 和 S_V 都是空串,所以欲证明这个系统是(完美)零知识的,只需证明对所有的充分大的 n 和所有的 $(x,y)\in\text{SQNR}(n)$,有 $\text{View}_V^{(P)}(1^n,(x,y))=S(1^n,(x,y))$。其中 $\text{View}_V^{(P)}(1^n,(x,y))=\{(\sigma,\text{proof}):\sigma\leftarrow\{0,1\}^{n^3};\text{proof}\leftarrow P_2(1^n,\sigma,(x,y))\}$。亦即证明由 S 输出的随机变量和由 V_2 看到的随机变量是完全一样的,这样,这两个随机变量不能被任何观察者(Observer)或区分者(Distinguisher)区别开来。

事实上,易知 σ 随机地分布在所有的 n^3b 长的串上。另外,如果 $\sigma_i\in J_x^{+1}$,那么对应的 s_i 是 σ_i 的一个随机的(x,y)根。这样,关于输入(x,y)和 σ,s_i 属于 S 的输出的概率和属于从证明者 P 发送给验证者 V 的证明的概率是一样的。故 $\text{View}_V^{(P)}(1^n,(x,y))=S(1^n,(x,y))$。

目前已有不少文献讨论了非交互零知识证明系统,主要讨论其实现问题,文献[5]中作了详尽的论述。文献[21]中解决了文献[4]中的两个公开问题,文献[22]中也独立地解决了文献[4]中的第一个公开问题。文献[21]中证明了任何单向置换能代替二次剩余并证明了对实现有界的非交互零知识证明单向置换是充分的,但证明者需要指数计算能力。文献[23]中指出了文献[21]中的漏洞,并说明了如何通过一个附加的证书来修改这个漏洞。文献[24]中说明了非交互零知识证明和不变数字签名的等价性。如果任何单向函数存在,文献

[25]中指出，在一个交互的预处理阶段后，任何充分短的定理能被非交互地、零知识地证明，而文献[26]中指出，在通过执行一个健忘传输协议的预处理阶段后，任何定理能被非交互地、零知识地证明。

5.2.3 公开可验证的非交互零知识证明

由证明者 P 所提供的一个定理的证明，关于它的验证有以下两种可能性。

(1) 定理的证明被直接提供给一个特定的验证者 V 而且只有 V 才能验证。

(2) 定理的证明能被系统中的任何用户验证。

在后一种情况下，称证明是公开可验证的。

我们说一个非交互零知识证明系统是公开可验证的，如果这个系统的证明者所提供的证明是公开可验证的。

公开可验证的非交互零知识证明系统的重要性在于它可以应用于数字签名和消息认证等密码协议之中，证明的公开可验证性将对应于任何人能检验签名。文献[27]、[28]中提出了实现公开可验证的非交互零知识证明系统的一些方法，这些系统都允许在非交互地、零知识地证明定理之前，许多证明者和验证者无需相互交互，但文献[3]～[5]中的系统限制证明者和验证者共享一个参考串来非交互地进行零知识证明，这样，如果许多用户希望互相证明定理，那么它们中的每对都不得不共享一个参考串，参考串的数量随用户数量的增加而迅猛地增加，这是不切合实际的。

在需要执行大量安全协议的大型网络中，非交互零知识证明是十分有用的。这是因为在这些协议中零知识是十分关键的，同时交互需要花更大的代价，一个典型安全协议的内部计算能在几秒钟内完成，而交换电子邮件所花的时间是它的上百倍。这样，在安全协议中，非交互零知识证明也许被用来节省昂贵的通信圈。文献[29]中讨论了非交互零知识证明在公钥密码系统中的应用，文献[23]中对此也作了进一步讨论。文献[27]中介绍了非交互零知识证明在数字签名、消息认证和识别号的无记忆分配等方面的一些应用。

5.3 Sigma 协议

自从证据不可区分这个概念提出以来，它便成为构造零知识证明/论证系统最为广泛使用的基本工具，绝大多数零知识证明/论证系统都是从一个 3 轮证据不可区分的证明/论证系统转化而来。在很多情况下，对 NP 语言 L 的满足某些特定性质的零知识论证系统一般通过以下方式来构造：先对 L 构造一个 3 轮证据不可区分的论证系统，然后利用一个组合器来把它转化成满足要求的目标论证系统。通常，组合器是由一些对某些具体的(数论)问题(可能独立于语言 L)的 3 轮证据不可区分论证系统构成的，这样，转化后的零知识协议的效率就严重依赖于组合器中对于这些具体的数论问题的 3 轮证据不可区分论证系统的效率。在这种背景下，一些常见的 3 轮证据不可区分论证系统的某些特殊性质就被抽象出来成为一个非常有用的模块，这就是 Sigma 协议，记为 Σ 协议[30]。

Σ 协议是一个具有特殊合理性的诚实验证者零知识协议。我们将看到对于一些具体的语言，能绕过 Cook-Levin 定理，即无需先把它归约到 NP 完全语言，直接构造非常高效的 Σ 协议。

一个对语言 L 的 Σ 协议(P,V)是一个满足下面两个条件的 3 轮公开掷币的证明/论证系统。

(1) 特殊合理性：记(P,V)中第 1、2、3 轮消息分别为(a,e,z)(一般称消息 e 为挑战)。给定两个形如(a,e,z)和(a,e',z')的可接受的副本，只要 $e\neq e'$，便可计算出公共输入 $x\in L$ 的证据 w。

(2) 特殊诚实验证者零知识：对于 $x\in L$，给定一个随机挑战 e，存在一个概率多项式时间模拟器 S，输出一个可接受的副本(a,e,z)，且它与诚实证明者和诚实验证者之间的真实交互所产生的副本是完美(统计、计算)不可区分的。

不难验证协议 5.1 和协议 5.2 都满足 Σ 协议的两个条件，于是便得到了以下定理。

定理 5.4　如果存在单向置换，则对任何 NP 语言存在 Σ 协议。

直觉上，由于能够从对任意两个不同的挑战的响应中提取出相应的证据，所以 Σ 协议是合理性错误为 $\frac{1}{2^{|e|}}$ 的知识的证明/论证系统。严格的证明略显繁琐，推荐读者参考 Damgard 的讲义[30]。

定理 5.5　Σ 协议是一个合理性错误为 $\frac{1}{2^{|e|}}$ 的知识的证明/论证系统。

虽然 Σ 协议没有要求证据不可区分这一非常有用的性质，但这一性质可以通过对 Σ 协议的异或合成来获得：给定对两个语言 L_0、L_1 的 Σ 协议 Σ_0 和 Σ_1，可以构造一个对语言 $L\triangleq L_0\vee L_1$ 的 Σ_{OR} 协议，它满足证据不可区分性。

协议 5.3

公共输入：$(x_0,x_1)\in L$(即证明 $x_0\in L_0$ 或 $x_1\in L_1$)。

P 的私有输入：w 使得$(x_b,w)\in R_{L_b}$，$b=0$ 或 1。

$P\to V$：按照协议 Σ_b 利用私有输入 w 计算这个协议的第一轮消息 a_b，随机选择一个挑战 e_{1-b}，利用模拟器 S 计算协议 Σ_{1-b} 的一个可接受的交互副本$(a_{1-b},e_{1-b},z_{1-b})$，然后它按随机顺序发送 a_b 和 a_{1-b}。

$V\to P$：发送一个随机比特 e。

$P\to V$：置 $e_b=e\oplus e_{1-b}$，把 e_b 当成 Σ_b 协议的挑战，利用 w 计算 Σ_b 的最后一轮消息 z_b，发送 e_b、e_{1-b}、z_b 和 z_{1-b}。

V 的输出：V 接受当且仅当 $e_b=e\oplus e_{1-b}$，并且两个副本(a_b,e_b,z_b)、$(a_{1-b},e_{1-b},z_{1-b})$都是可接受的。

命题 5.1　如果 Σ_0 和 Σ_1 都是具有特殊诚实验证者完美(统计、计算)零知识的 Σ 协议，则对于来自 $x_0\in L_0$ 的证据和 $x_1\in L_1$ 的证据，协议 5.3 是完美(统计、计算)证据不可区分的。

证明　这里仍然使用混合论证的办法来证明。考虑以下证明者策略。

证明者 HP：HP 执行证明者 P 的策略，但是它在执行协议 Σ_{1-b} 时使用 $x_{1-b}\in L_{1-b}$ 的证据，而不是利用模拟器 S 来预先计算一个副本。

分别记 $x_0\in L_0$ 和 $x_1\in L_1$ 的证据为 w_0 和 w_1。由于 Σ_0 和 Σ_1 都满足特殊诚实验证者完美(统计、计算)零知识性，所以对任意 b，分布$(\mathrm{PH},V)(x_0,x_1)$和分布$(\mathrm{PH},V)(x_0,x_1)$是完美(统计、计算)不可区分的，进而有$(P(w_0),V)(x_0,x_1)$和$(P(w_1),V)(x_0,x_1)$是完美(统

计、计算)不可区分的。

虽然定理5.3断言对任意的NP语言L都能构造Σ协议(给定语言L的一个实例x,先把它转化成一个汉密尔顿图G,然后对G构造一个Σ协议),但这样的协议通常计算效率非常低。下面给出几个对具体数论问题的非常高效的Σ协议。

第一个是针对离散对数问题的协议。设n为安全参数,p、q为两个素数使得$p=2q+1$,$|q|=n$,G_q为Z_p^*的一个阶为q的子群,g是G_q的一个生成元。给定公共输入(g,h,p,q),这里h和g为G_q中两个元素,$h=g^w \bmod p$。现在假设证明者知道离散对数w,他向验证者证明他知道w。

协议 5.4

公共输入:(g,h,p,q)。

P的私有输入:w使得$h=g^w \bmod p$。

$P\to V$:随机选择一个$r\leftarrow Z_q$,计算$a=g^r \bmod p$,并发送a。

$V\to P$:发送一个随机比特串$e\leftarrow Z_q$。

$P\to V$:发送$z=(r+ew) \bmod q$。

V的输出:V接受当且仅当$g^z\equiv ah^e \bmod p$。

很容易验证协议5.4是一个Σ协议。给定挑战随机串e,可以随机选择一个z,计算$a=g^z/h^e$。另外,如果某个证明者能够回答两个不同的挑战e和e',就可以计算h的离散对数$w=(z-z')/(e-e')$,这里z'是证明者对挑战e'的一个正确的回答。

第二个是针对二次剩余问题的协议。设n为安全参数,设$N=pq$,p、q为两个素数使得$p=2q+1$,$|q|=n$。令J_N为Z_N^*中所有Jacobi符号为1的元素组成的子群,QR_N为Z_N^*中所有二次剩余组成的子群。给定公共输入$s\in J_N$与N,现在假设证明者知道s的一个平方根$w(w^2\equiv s(\bmod N))$,他向验证者证明$s\in QR_N$。

协议 5.5

公共输入:$s\in J_N$与N。

P的私有输入:w使得$w^2\equiv s(\bmod N)$。

$P\to V$:随机选择一个$r\leftarrow Z_N$,计算$a=r^2 \bmod N$,并发送a。

$V\to P$:发送一个随机比特$e\leftarrow\{0,1\}$。

$P\to V$:如果$e=0$,发送$z=rw \bmod N$;如果$e=1$,发送r。

V的输出:$e=0$时,接受当且仅当$as\equiv z^2(\bmod N)$;$e=1$时,接受当且仅当$a=r^2 \bmod N$。

读者可以验证协议5.5也符合Σ协议的要求。

对于某些语言的Σ_{OR}协议,除了具有证据不可区分性,还具有另一个重要的性质,即证据隐藏。下面以对离散对数问题的Σ_{OR}协议为例来介绍这一性质。

协议 5.6

公共输入:(g,h_0,h_1,p,q)。

P的私有输入:w_b使得$h_b=g^{w_b} \bmod p$。

$P\to V$:随机选择一个$r\leftarrow Z_q$,$e_{1-b}\leftarrow Z_q$和$z_{1-b}\leftarrow Z_q$,计算$a_b=g^r \bmod p$和$a_{1-b}=g^{z_{1-b}}/h_{1-b}^{e_{1-b}} \bmod p$并按随机顺序发送$a_b$,$a_{1-b}$。

$V\to P$:发送一个随机比特串$e\leftarrow Z_q$。

$P \to V$：计算 $e_b = e \oplus e_{1-b}$，发送 $z_b = (r + e_b w) \bmod q$ 和 z_{1-b}。

V 的输出：接受当且仅当 $e_b = e \oplus e_{1-b}$ 且 $g^{z_b} \equiv a_b h_b^{e_b} \pmod p$，$g^{z_{1-b}} \equiv a_{1-b} h_{1-b}^{e_{1-b}} \pmod p$。

在分析协议 5.6 之前，首先给出困难关系及证据隐藏这两个概念。

如果一个由语言 L 所确定的关系 R_L 满足下面两个条件，就说它是困难的。

(1) 实例易生成：存在一个 PPT 算法 G，给定 1^n 作为输入，输出对 $(x, w) \in R_L$，这里 $|x| = n$。

(2) 证据难计算：对所有的 PPT 算法 A，给定一个由算法 G 随机输出对 (x, w) 中的实例 x，A 计算出某个 w' 使得 $(x, w') \in R_L$ 的概率是可忽略的。

在标准的离散对数假设下，由离散对数问题所确立的关系

$$R_L = \{((g, h, p, q), w) \mid h = g^w \bmod p\}$$

就是一个困难关系。

设 (P, V) 是一个对语言 L 的证明/论证系统，并且语言 L 所确定的关系 R_L 是困难的。如果对于任意 PPT 验证者 V'，它在与诚实证明者交互后输出实例 x 的一个有效证据 w 的概率是可忽略的，就说 (P, V) 是证据隐藏的。

现在来验证协议 5.6 是证据隐藏的。

命题 5.2 在离散对数假设下，协议 5.6 是证据隐藏的。

证明 假设存在一个 PPT 验证者 V' 在和诚实的证明者交互后能以不可忽略的概率输出实例 (g, h_0, h_1, p, q) 的某个证据 w_i（对于某个 $i \in \{0, 1\}$ 有 $h_i = g^{w_i} \bmod p$）。考虑以下算法 M。

算法 M

输入：(g, h, p, q)。

随机选择 $b \leftarrow \{0, 1\}$，$w_b \leftarrow Z_q$，计算 $h_b = g^{w_b} \bmod p$，置 $h_{1-b} = h$。以 (g, h_0, h_1, p, q) 为公共输入，利用 w_b 为证据扮演证明者与 V' 交互。输出 V' 的输出。

观察到协议 5.6 是（完美）证据不可区分的。如果在真实的交互中，验证者 V' 在交互后能以不可忽略的概率 $p(n)$ 输出实例 (g, h_0, h_1, p, q) 的某个证据 w_i，则算法 M 将以 $p(n)/2$ 的概率输出 $w = w_{1-b}$ 使得 $h = g^w \bmod p$（如果输出 $w = w_b$ 的概率高于 $p(n)/2$，则能利用验证者 V' 来区分诚实证明者在执行协议 5.6 时所用的证据），这样就打破了离散对数假设。

5.4 常数轮零知识协议

一般来讲，构造的零知识协议要么具有较大的（不可忽略的）合理性错误，要么不是常数轮（如对汉密尔顿图的 3 轮零知识证明系统顺序合成后产生的协议）。本节将介绍怎样构造对 NP 语言的具有可忽略合理性错误的常数轮零知识协议。每当讲到常数轮时，自然隐含该协议的合理性错误是可忽略的。

下面将从几种不同的途径来构造常数轮零知识协议，重点使用两个技巧：一个是陷门承诺方案；另一个是 FLS 技巧。整体上讲，这些途径遵循着一个基本框架：先让验证者生成某些陷门信息并向证明者证明它知道这些信息，然后证明者向验证者证明公共输入的真实性。这里无论是证明者还是验证者使用的证明系统都不是零知识的，它们也不需要满足零知识性（有时满足证据不可区分性即可），此外，如果模拟器能得到验证者生成的陷门信

息,它就能在无需知道公共输入的证据情况下直接模拟证明者的证明。

首先来介绍使用陷门承诺方案来构造常数轮零知识协议的途径。

我们将用到一个由具有特殊诚实验证者完美零知识的 Σ_{OR} 协议转化而成的陷门承诺方案。注意到 Σ_{OR} 协议本身也是 Σ 协议。给定一个对困难关系 R_L 的具有特殊诚实验证者完美零知识的 Σ 协议,下面给出由此协议到承诺方案的一般转化方法。

基于特殊诚实验证者完美零知识的 Σ 协议的承诺方案如下。

(1) 建立公共参数:给定 1^n 作为输入,接收者运行 PPT 算法 G,得到 $(x,w)\in R_L$,将 x 发送给承诺者。

(2) 承诺阶段:给定被承诺值 e,承诺者把 x 和 e 输入对应 Σ 协议的模拟器 M,得到一个随机副本 (a,e,z),把 a 作为承诺发送给接收者。

(3) 打开阶段:承诺者把 (e,z) 发送给接收者。接收者验证 (a,e,z) 是否为一个可接受副本。

由于模拟器 M 的输出与真实交互完美不可区分,所以 M 产生的第一条证明者消息 a 与那些由诚实证明者产生的能回答所有挑战的第一条证明者消息服从同一分布。这也就说明了上述承诺方案具有完美隐藏性。另一方面,如果承诺者能把同一个承诺 a 打开成两个不同的 e 和 e',则根据 Σ 协议的特殊合理性,可以计算出 x 的一个证据 w,而这违背了关系 R_L 的困难性。

容易把上述承诺方案改成一个陷门承诺方案。在建立公共参数阶段,让接收者生成 $(x,w)\in R_L$ 后向承诺者证明它知道 x 的一个证据 w。通常这样的证明本身也要求是零知识的。注意到这里的目标是构造常数轮零知识协议,而陷门承诺方案本身只是一个构成部件,所以要求在这样一个部件中嵌入一个常数轮零知识证明就显得不合理。

基于陷门承诺方案的对汉密尔顿图的常数轮零知识论证系统构造如下。

验证者(在陷门承诺方案中扮演接收者的角色)首先生成两个离散对数实例 (h_0,w_0) 和 (h_1,w_1) 使得 $h_b=g^{w_b} \bmod p, b=0,1$,然后利用一个证据不可区分协议(即 Σ_{OR} 协议)向证明者证明他知道 w_0 或 w_1。在第二阶段,证明者利用对汉密尔顿图的 3 轮证明系统(协议 5.2)向验证者证明图的汉密尔顿图,在这个证明中,证明者所用的承诺方案是由协议 5.6 按上述的方法转化而来的承诺方案。

协议 5.7

公共输入:汉密尔顿图 $G=(V,E)$,$|V|=n$。

P 的私有输入:图 G 的汉密尔顿图 H。

第一阶段:验证者 V 随机选择素数 p,q 使得 $p=2q+1$,$|q|=n$。随机选择 $w_0\leftarrow Z_q$,$w_1\leftarrow Z_q$ 和 $b\leftarrow\{0,1\}$。计算 $h_0=g^{w_0} \bmod p$ 和 $h_1=g^{w_1} \bmod p$。V 发送 (g,h_0,h_1,p,q),调用对此实例的 Σ_{OR} 协议(协议 5.6)向证明者 P 证明它知道 w_0 或 w_1。

第二阶段:证明者 P 调用协议 5.2 向验证者 V 证明图 G 是汉密尔顿图。在调用协议 5.2 时,证明者在第一步使用以下方式产生的承诺方案:①将协议 5.6 的挑战 e 的长度限制为 1b(因为协议 5.2 中证明者逐比特进行承诺);②将修改后的协议 5.6 按照上述的方式转化成一个比特承诺方案。

注:

(1) 在第一阶段,验证者需要选择两个离散对数实例,而不是一个。这样做是必需的,

否则将无法证明整个系统的合理性。这是因为用来证明合理性的归约算法(它扮演验证者的角色)需要利用欺骗证明者的能力来打破离散对数假设,但利用欺骗证明者的能力反过来也需要这个归约算法知道一个 w_b。在这里,归约算法可以生成两个离散对数实例,拥有其中一个 w_b,它仍有机会利用欺骗证明者的能力来得到另一个实例的离散对数,进而打破离散对数假设,证明合理。

(2) 可以合理安排协议 5.7 的消息调度将轮数降低至 4 轮。如将第二阶段协议 5.2 的第一轮证明者消息和第一阶段协议 5.6 的第二轮消息合并为整个大协议的第二轮消息,并将第一阶段协议 5.6 的第三轮消息和第二阶段协议 5.2 的第二轮消息合并为整个大协议的第三轮消息。

现在来证明以下定理。

定理 5.6 如果离散对数假设成立,则协议 5.7 为完美零知识论证系统。

证明 完全性显然。下面证合理性。

把协议 5.7 的合理性归约到离散对数假设上。

假设公共输入图 G 不是汉密尔顿图,并且存在一个 PPT 欺骗证明者 P^* 使得验证者 V 以不可忽略的概率 p 接受。考虑以下离散对数算法。

算法 B

输入:(g,h,p,q)。

(1) 随机选择 $b \leftarrow \{0,1\}$,$w_b \leftarrow Z_q$,计算 $h_b = g^{w_b} \bmod p$,置 $h_{1-b} = h$。发送 (g,h_0,h_1,p,q),并利用 w_b 作为证据调用协议 5.6 向证明者 P^* 证明它知道 w_b。

(2) 当接收到第二阶段协议 5.2 的最后一条证明者消息 z,如果它不是一个正确的回答,则停机输出 $\perp$;如果它是一个正确的回答,设对应的协议 5.2 的可接受副本为 (a,e,z),重新发送一个新的随机挑战 e',$e' \neq e$,直至得到另一个可接受副本 (a,e',z')。

(3) 找到消息 a(它由 n 个比特承诺矩阵组成)中某个位置上的承诺,它在最后 z 与 z' 中被打开成不同的比特(注意到当图 G 不是汉密尔顿图时,必然存在这样的位置,否则我们能由 (a,e,z) 与 (a,e',z') 计算出 G 的汉密尔顿图)。设这个位置的承诺为 (a_0,a_1)(协议 5.6 的第一条消息),它在 z 中被打开成 0,其打开形式为 (e_0,e_1,z_0,z_1) 且有 $0 = e_0 \oplus e_1$,它在 z' 中被打开成 1,这个打开形式为 (e_0',e_1',z_0',z_1') 且有 $1 = e_0' \oplus e_1'$。从这两个不同的打开中找到两个协议 5.4(对离散对数的 Σ 协议,注意到协议 5.6 是协议 5.4 的异或合成)的可接受副本,(a_c,e_c,z_c) 和 (a_c,e_c',z_c'),$e_c \neq e_c'$,$c=0$ 或 1,计算 h_0 或 h_1 的离散对数,并输出。

如果 P' 使得验证者 V 接受的概率 p 是不可忽略的,则读者可以验证上述算法将在期望多项式时间内停机输出。此外,由于 G 不是汉密尔顿图,算法 B 将以概率 p 输出一个(h_0 或 h_1 的)离散对数。并且,注意到第一阶段协议是证据完美不可区分的,可知算法 B 将以概率 $\frac{p}{2}$ 输出 h 的离散对数。这就打破了离散对数假设。

完美零知识性。构造一个模拟器来证明完美零知识性。基本想法是让模拟器在执行协议第一阶段时提取出 h_0 或 h_1 的离散对数,然后完全按照诚实证明者的策略来执行第二阶段子协议(协议 5.2)的第一轮证明者消息(设为 $\bar{a}$),当收到这个子协议的挑战,它利用 h_0 或 h_1 的离散对数把 $\bar{a}$ 中的承诺打开成能回答这一挑战的值(注意到它在产生消息 $\bar{a}$ 时使用的是陷门承诺方案),产生一个可接受的副本。

模拟器 S

(1) 为 V' 选择随机带 r。

(2) 在第一阶段子协议中执行诚实证明者策略，发送随机挑战 e。

(3) 若 V' 停机或接收到的第一阶段协议 5.6 的最后一条证明者消息 z 不是一个正确的回答，则停机输出 V' 的输出；如果它是一个正确的回答，设对应的协议 5.7 的可接受副本为 (a,e,z)，重新发送一个新的随机挑战 e'，$e' \neq e$，直至得到另一个可接受副本 (a,e',z')。

(4) 从上面两个副本中计算出 h_0 或 h_1 的离散对数。然后完全按照诚实证明者的策略来执行第二阶段子协议(协议 5.2)的第一轮证明者消息 $\bar{a}$，当收到这个子协议的挑战，它利用 h_0 或 h_1 的离散对数把 $\bar{a}$ 中的承诺打开成能回答这一挑战的值，发送第二阶段子协议的最后一轮消息。

(5) 输出 V' 的输出。

设 V' 在第(3)步正确回答第一阶段子协议挑战的概率为 p，模拟器 S 单次执行协议 5.7 的时间为多项式 $q(n)$，则 S 的期望运行时间为

$$pq(n) + \left(p\sum_{n=1}^{\infty} np(1-p)^{n-1}\right)q(n) = \text{poly}(n)$$

分别记 V' 在真实交互和模拟中正确回答第一阶段子协议挑战这一事件为 E_{real} 和 E_{sim}。观察到

$$\Pr[E_{\text{real}}] = \Pr[E_{\text{sim}}] = p$$

显然，对于任意可能的 V' 的输出 β，有

$$\Pr[S = \beta \mid E_{\text{sim}}] = \Pr[(P,V')(G) = \beta \mid E_{\text{real}}]$$

注意到在第二阶段子协议执行中，证明者所使用的陷门承诺方案是完美隐藏的，所以

$$\Pr[S = \beta \mid \overline{E_{\text{real}}}] = \Pr[(P,V')(G) = \beta \mid \overline{E_{\text{sim}}}]$$

由上述讨论可知，S 的输出和 V' 在真实交互中的输出服从同一分布。

第二种构造途径利用优雅而简洁的 FLS 技巧。与第一种途径相比，它让模拟器更加直接地利用第一阶段得到的陷门信息 τ：在第二阶段证明中，证明者利用证据不可区分证明系统证明以下断言：

“我知道图 G 的一个汉密尔顿图或陷门信息 τ”

其具体描述如下。

协议 5.8

公共输入：汉密尔顿图 $G=(V,E)$，$|V|=n$。

P 的私有输入：图 G 的汉密尔顿圈 H。

第一阶段：验证者 V 选择一个单向置换 f，随机选择 $x_0 \leftarrow \{0,1\}^n$，$x_1 \leftarrow \{0,1\}^n$。计算 $y_0 = f(x_0)$ 和 $y_1 = f(x_1)$。发送 (f, y_0, y_1)，调用一个证据不可区分证明系统向证明者 P 证明它知道 x_0 或 x_1。

第二阶段：证明者 P 调用一个证据不可区分证明系统向验证者 V 证明他知道图 G 的汉密尔顿圈 H 或 x_0 或 x_1。

这里有两种方式来实现协议 5.8 中证据不可区分证明系统。第一种就是前面介绍的 Σ_{OR} 协议。第二种方式是先把所要证明的“或”断言转化成一个汉密尔顿图 G'，然后调用协议 5.2 来证明图 G' 是一个汉密尔顿图。

可以用类似于分析协议 5.7 的方式来分析协议 5.8。读者可以证明协议 5.8 也是一个零知识论证系统。

定理 5.7 如果存在单向置换,则对任意 NP 语言存在 4 轮零知识论证系统。

上面两种方式构造的协议都是论证系统:如果允许敌意的证明者具有无限计算能力,它就能计算出这些协议中验证者所产生的陷门信息,进而能使诚实验证者接受一个非汉密尔顿图。

下面给出一个 5 轮的计算零知识证明系统。它的一个基本想法是让验证者把它在 3 轮证据不可区分证明系统(如协议 5.2)中的挑战预先承诺起来。这样做的好处就是当模拟器看到这个挑战后,它能按照这个挑战重新生成这个 3 轮子协议的第一轮证明者消息,而不必担心生成新的消息后验证者会改变挑战(这由承诺方案的计算绑定性所保证)。

协议 5.9

公共输入:汉密尔顿图 $G=(V,E)$,$|V|=n$。

P 的私有输入:图 G 的汉密尔顿圈 H。

第一阶段:验证者和证明者交互产生一个完美隐藏的承诺方案 Com。验证者选择一个随机数 r 来承诺挑战 e,$c=\text{Com}(e,r)$。发送 c。

第二阶段:证明者 P 调用协议 5.2 向验证者 V 证明他知道图 G 的汉密尔顿圈 H。在这个执行中,当验证者需要发送挑战时,它发送(e,r)。证明者验证挑战 e 是否是承诺 c 的一个有效的承诺值,如果是,则回答挑战 e;如果不是,则停机。

对这个协议的分析在一些标准的教科书中可以找到。注意到在无爪(Claw-free)单向置换存在的假设下可以构造 2 轮的完美隐藏的承诺方案,于是有以下定理。

定理 5.8 如果存在无爪单向置换,则对任意 NP 语言存在 5 轮计算零知识证明系统。

5.5 小结

零知识证明理论与方法在密码算法和安全协议的设计与分析中占有十分重要的地位,特别是在安全协议的设计与分析中尤为重要。本章较系统地介绍了零知识证明理论与方法的一些基本知识。构造了对 NP 语言的零知识证明/论证系统,并介绍了零知识协议的一些有着广泛应用的变体,如证据不可区分协议[31]、Sigma 协议[30]、证据隐藏协议[31]以及一些重要的技巧。另外还讨论了怎样去掉交互性来构造非交互零知识证明系统。

由于篇幅的原因,并没有涉及一些近年发展起来的高级课题,如并发合成,可重置零知识、普适合成等。对于非交互零知识,也没有介绍在使用单个公共参考串情形下构造对多个定理的零知识证明[32]。另外,怎样构造带有多项式时间证明者的非交互零知识证明也是一个非常有挑战性的问题,如目前还不知道在只假设单向函数存在的条件下如何来构造带有多项式时间证明者的非交互零知识证明系统。非交互的、信息论意义下安全的可验证的秘密共享方案的研究可参阅文献[33]。

关于零知识证明理论与方法的一些发展和应用在各章节中已作了一些介绍,感兴趣的读者可在计算机科学基础研讨会(Foundation Of Computer Science (FOCS) Conference)、计算理论专题研讨会(Symposium on the Theory Of Computing (STOC) conference)、各种密码会议(如 Crypto conference、Eurocrypt conference 等)的论文集中找到很多有关这个主

题的论文。

关于零知识证明理论与方法的综述性文章和参考书可参阅文献[34]～[38]。近年来，我国学者在零知识证明理论研究方面也取得了一系列国际领先的研究成果[39～43]，如邓燚博士在双重可重置猜想证明方面的研究成果[42]。本章在写作过程中得到了邓燚副研究员的大力支持，他提供了大量的相关材料，作者在此表示衷心的感谢。

参考文献

[1] Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof systems, Proceedings of the 17th ACM Symposium on the theory of Computing, 1985, 291～304.

[2] Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof systems, SIAM J. Comput, 18(1989), 186～208.

[3] Blum M, Feldman P, Micali S. Non-interactive zero-knowledge proof systems and applications, in Proc. 20th Annual ACM Symposium on theory of Computing, Chincago, IL, 1988, 103～112.

[4] De Santis A, Micali S, Persiano G. Non-interactive zero-knowledge proof systems, Advances in Cryptology-Crypto'87, Springer-Verleg, Berlin, New York, 1987, 52～72.

[5] Blum M, De Santis A, Micali S. Non-interactive zero-knowledge, SIAM J. Comput., 20 (1991), 1084～1118.

[6] Feige U, Fioot A, Shamir A. Zero knowledge proofs of identity, Proceedings of STOC, 1987, 210～217. (J. Of Cryptology, Vol. 1, 1988, 77～94)

[7] Quisquater J J, Guillou L, Berson T. How to explain zero-knowledge protocols to your children, Advances in Cryptology-Crypto'89, Springer-Verleg, 1990, 628～631.

[8] Balcazar J, Diaz J, Gabarro J. Structural Complexity I, II, EATCS Monographs on theoretical computer science, springer-Verlay, Berlin, 1988, 1991.

[9] Sakurai K, Itoh T. On the discrepancy between serial and parallel of zero-knowledge protocols, Advances in Cryptology-Crypto'92, Springer-Verlay, 1993, 264～259.

[10] Ben-or M, Goldwasser S, Kilian J. Multi-Prover interactive proofs: How to remove intractability assumptions STOC 1988, 113～131.

[11] Brassard G, Chaum D, Crépeau C. Minimum disclosure proofs of knowledge, Journal of Computer and systems Science, 37(1988), 156～189.

[12] Goldreich O, Micall A, Wigdeson A. Proofs that yield but their Validit or all languages in NP have zero-knowledge proof systems, Journal of the ACM, 38(1991), 691～729.

[13] Stinson D R. Crytography-Theory and practice, CRC press, 1995.

[14] Galil Z, Haber S, Yung M. A private interactive test of a Boolean predicate and minimum-knowledge public key Cryptosystems, proceeding of FOCS 1985, 360～371.

[15] Bellare M, Goldreich O. On defining Proofs of knowledge, Advances in Cryptology-Crypto'92, Springer-Verlag, Berlin, 1993, 390～420.

[16] Feige U, Shamir A. Zero-knowledge proof of knowledge in two rounds, Advances in Cryptology-Crypto'89, Springer-Verlay, Berlin, 526～544.

[17] Tompa M, Woll H. Random self-reducibility and zero-knowledge interactive proofs of possession of information, Proceedings of the 28th Symposium on Foundations of Computer Science, 1987, 472～482.

[18] De Santis A, Persiano G. The Power of Preprocessing in zero-knowledge proofs of knowledge, J. Cryptology, Vol. 9, No. 3, 129～148, 1996.

[19] De Santis A, Persiano G. Zero-knowledge proofs of knowledge without interaction, proceeding of the 33rd Symposium on Foundations of Computer Science, 1992, 427～437.

[20] Rabin M. Probabilistic algorithm for testing primality, J. Number Theory, 12(1980), 128～138.

[21] Feige U, Lapidot A, Shamir A. Multiple non-interactive zero-knowledge proofs based on a single random string, in Proc. 31st Annual IEEE Symposium on Foundations of Computer Science, St. Louis, Mo, 1990, 308～317.

[22] De Santis A, Yung M. Cryptographic applications of the non-interactive metaproof and many-prover systems, Advances in Cryptology-Crypto'90, Springer-Verlay, Berlin, NewYork, 1990, 366～377.

[23] Bellare M, Yung M. Certifying Cryptographic Tools: The case of trapdoor permutations, Advances in Cryptology-Crypto'29, Springer-Verlay, Berlin, New York, 1992, 442～460.

[24] Goldwasser S, Ostrovsky R. Invariant signatures and non-interactive zero-knowledge proofs are equivalent, Advances in Cryptology-Crypto'92, Springer-Verlay, Berlin, New York, 1992, 228～245.

[25] De Santis A, Micali S, Persiano G. Non-interactive zero-knowledge proof-systems with preprocessing, Advances in Cryptology-Crypto'88, Springer-Verlay, Berlin, New York, 1988, 269～283.

[26] Kilian J, Micali S, Ostrovsky R. Minimum resource zero-knowledge proofs, Advances in Cryptology-Crypto'89, Springer-Verlay, Berlin, New York, 1989, 545～546.

[27] Bellare M, Goldwasser S. New paradigms for digital signatures and message authentication based on non-interactive zero-knowledge proofs, Advances in Cryptology-Crypto'89, Springer-Verlay, Berlin, New York, 1989, 194～211.

[28] Bellare M, Micali S. Non-interactive obvious transfer and applications, Advances in Cryptology-Crypto'89, Springer-Verlay, Berlin, New York, 1989, 547～559.

[29] Naor M, Yung M. Public-key cryptosystems Provable secure against choosen ciphertext attack, in proc. 22nd Symposium on theory of computing, Baltimore, MD, 1990, 427～437.

[30] Damgard I. On Sigma-Protocol. Lecture Notes, available on http://www. daimi. au. dk/～ ivan/Sigma. pdf.

[31] Feige U, Shamir A. Witness Indistinguishability and Witness Hiding Protocols. In Proc. of ACM STOC 1990, 416～426.

[32] Feige U, Lapidot D, Shamir A. Multiple NonInteractive Zero Knowledge Proofs Under General Assumptions. SIAM J. Comput. 29(1): 1～28 (1999).

[33] Pedersen T. Non-Interactive and information-theoretic Secure Verifiable secret sharing. In: Feigenbum(Ed.). Advances in Cryptology-Crypto'91, LNCS No. 576. Berlin: Springer-Verlag Heidelberg, 1992. 129～140.

[34] Simmons G J. Contemporary Cryptology, IEEE press, 1991.

[35] 朱洪,吴京.零知识证明浅介,密码与信息,1992,No. 4,1～38.

[36] 冯登国.非交互零知识证明及其密码应用,密码与信息,1996,No. 2,14～22.

[37] 冯登国,裴定一.密码学导引.北京:科学出版社,1999.

[38] Goldreich O. Foundation of Cryptography-Basic Tools. Cambridge University Press, 2001.

[39] Yunlei Zhao, Xiaotie Deng, Chan H. Lee, Hong Zhu. Resettable Zero-Knowledge in the Weak Public-Key Model. EUROCRYPT 2003: 123～139.

[40] Yi Deng, Dongdai Lin. Instance-Dependent Verifiable Random Functions and Their Application to Simultaneous Resettability. EUROCRYPT 2007: 148～168.

[41] Yi Deng, Giovanni Di Crescenzo, Dongdai Lin, Dengguo Feng: Concurrently Non-malleable Black-Box Zero Knowledge in the Bare Public-Key Model. CSR 2009: 80～91.

[42] Yi Deng, Vipul Goyal, Amit Sahai. Resolving the Simultaneous Resettability Conjecture and a New Non-Black-Box Simulation Strategy. FOCS 2009: 251～260.

[43] Yunlei Zhao, Jesper Buus Nielsen, Robert H. Deng, Dengguo Feng: Generic yet Practical ZK Arguments from any Public-Coin HVZK Electronic Colloquium on Computational Complexity (ECCC)(162): (2005).

第6章 安全多方计算理论与方法

本章主要介绍安全多方计算(也称多方安全计算)的一般理论与方法,即安全多方计算的一般性结论,而不是解决具体问题的协议。重点突出一般性研究到实践的重要性,这主要体现在以下4个方面:一是刻画了分布式计算环境中的一些最基本的安全问题;二是从原理上明确了在既定的安全模型下哪些计算功能是可以安全实现的,哪些是不可行的;三是发展了设计分布式安全协议的一般性方法和技巧;四是给出了设计可应用于实际系统中的某些具体的方案和模块。

功能函数是安全多方计算中的一个重要概念。一个 m 元功能函数是指将 m 个输入映射到 m 个输出的随机过程。将 m 个输入映射到 m 个输出的函数是功能函数的特殊情形,也被称为确定性功能函数。换句话讲,功能函数是普通函数的随机化推广。可将功能函数 F 看作相应函数构成空间上的随机变量(即 F 等于 $f^{(i)}$ 的概率是 p_i),也可认为功能函数 F 随机选择一个串 r 并且以 $F'(r,x_1,\cdots,x_m)$ 作为输出,其中 F' 是将 $m+1$ 个输入映射到 m 个输出的函数。

一般地,将安全协议看做是将 m 个输入映射为 m 个输出的随机过程。安全多方计算的定义方式可以回溯到零知识证明和语义安全的定义方式,即称一个协议是安全的,如果敌手攻击实际协议所得与攻击理想模型所得相当。此处理想模型是指存在一个所有参与方共同信任的可信方,在可信方的帮助下计算出协议的功能函数。具体执行过程如下:每个参与方将自己的输入传输给可信方,可信方计算功能函数,将计算结果返回给相应的参与方。易见,理想模型是平凡的安全协议,那么敌手攻击实际协议所得与攻击理想模型所得相当,而理想模型是平凡的安全协议,攻击这样的平凡协议无所得,从而攻击实际协议也无所得,这样原来的协议就是安全的。

下面介绍定义安全多方计算安全模型所需要考虑的一些参数。

(1) 初始假设。除非特别声明,本章没有初始假设。在某些情况下假设每个参与方持有其他参与方的某些信息,如公钥等。

(2) 通信信道。本章关于信道的标准假设是敌手可以搭线窃听所有的通信信道。

(3) 计算能力限制。如无特别声明,本章讨论计算能力有界即概率多项式时间敌手。

(4) 敌手攻击能力。根据敌手入侵参与方的方式,可分为自适应和非自适应两种。自适应敌手在协议执行过程中,根据当前收集到的信息决定入侵哪个参与方;非自适应敌手在协议执行之前确定好要入侵的参与方集合。显然,自适应是比非自适应更为一般的攻击模型。另外,一个限制敌手攻击能力的参数是敌手控制参与方的方式,分成恶意和半诚实两种。恶意敌手不遵守协议指令,半诚实敌手遵守协议指令,只是收集并且记录信息。

(5) 安全性定义的限制。本章讨论的协议是“不公平”的,即不诚实方可以中断协议执行,这样某些诚实方得不到期望的输出,但是可以探测出协议被不诚实方中断。称这种安全性为允许中止的安全性。

本章考虑以下的协议运行环境即安全模型:不假设保密信道的存在,敌手攻击能力是

非自适应的、恶意的，并且是计算有界的。

6.1 安全两方计算

安全两方计算是安全多方计算的一类重要的特殊形式。定义安全两方计算要用到计算不可区分的概念，前面已经将区分器视作概率多项式时间算法或多项式规模电路族给出了计算不可区分的定义，这里将区分器视作多项式规模电路族来回顾一下不可区分的定义。把两个随机变量的可数集 $X=\{X_w\}_{w\in S}$ 和 $Y=\{Y_w\}_{w\in S}$ 称为计算不可区分的，如果对于任意多项式规模电路族 $\{C_n\}_{n\in N}$，任意正多项式 $p(\cdot)$，对于充分大的 n，每一个 $w\in S\cap\{0,1\}^n$，都有

$$|\Pr[C_n(X_w)=1]-\Pr[C_n(Y_w)=1]|<\frac{1}{p(n)}$$

6.1.1 半诚实模型中的安全两方计算

半诚实模型是指参与协议的双方中有一方是敌手，另一方是诚实方。此处的敌手只能施行半诚实攻击，即完全遵守协议的指令，只不过会记录协议运行中的信息和计算结果。半诚实模型是向恶意模型的过渡，为最终解决恶意模型中的安全两方计算问题提供思路和基础。

本节提出半诚实模型中安全性的两个等价定义。第一种定义方式是半诚实模型特有的一种简单方式，第二种定义方式是安全多方计算所采用的一般方式。这两种定义方式都基于模拟的方法。按照第一种定义方式定义的两方安全计算也称为两方保密计算。称一个两方协议 Π 可计算功能函数 f，如果协议关于输入对 (x,y) 的输出分布与功能函数 $f(x,y)$ 的输出分布相等。注意此时只考虑参与协议的双方都是诚实方的情形，协议的输出分布即协议计算功能函数还没有涉及安全性。

定义 6.1(两方保密计算) 设 $f:\{0,1\}^*\times\{0,1\}^*\rightarrow\{0,1\}^*\times\{0,1\}^*$ 是一个功能函数，$f_1(x,y)$ 和 $f_2(x,y)$ 分别是 $f(x,y)$ 的第一个和第二个分量。Π 是计算 f 的两方协议。定义第一方执行协议过程中的视图（也称观察）为 $(x,r,m_1,\cdots,m_t)$，记为 $\text{View}_1^{\Pi}(x,y)$，其中 r 表示第一方的内部掷币结果，m_i 表示第一方在协议执行过程中收到的第 i 个消息。同样地，第二方的视图为 $(y,r',n_1,\cdots,n_s)$，记为 $\text{View}_2^{\Pi}(x,y)$。第一方执行协议 Π 之后的输出记为 $\text{Output}_1^{\Pi}(x,y)$，第二方的输出记为 $\text{Output}_2^{\Pi}(x,y)$，因此，协议的整体输出记为 $\text{Output}^{\Pi}(x,y)=(\text{Output}_1^{\Pi}(x,y),\text{Output}_2^{\Pi}(x,y))$。当功能函数 f 是确定性函数时，称协议 Π 保密计算 f，如果存在概率多项式时间算法 S_1 和 S_2，使得

$$\{S_1(x,f_1(x,y))\}_{x,y\in\{0,1\}^*}\stackrel{c}{\equiv}\{\text{View}_1^{\Pi}(x,y)\}_{x,y\in\{0,1\}^*}\quad(6\text{-}1)$$

$$\{S_2(y,f_2(x,y))\}_{x,y\in\{0,1\}^*}\stackrel{c}{\equiv}\{\text{View}_2^{\Pi}(x,y)\}_{x,y\in\{0,1\}^*}\quad(6\text{-}2)$$

为简便起见，这里假定 $|x|=|y|$，其中 $\stackrel{c}{\equiv}$ 表示多项式规模电路族计算不可区分。

一般地，称协议 Π 保密计算 f，如果存在概率多项式时间算法 S_1 和 S_2，使得

$$\{(S_1(x,f_1(x,y)),f(x,y))\}_{x,y\in\{0,1\}^*}\stackrel{c}{\equiv}\{(\text{View}_1^{\Pi}(x,y),\ \text{Output}^{\Pi}(x,y))\}_{x,y\in\{0,1\}^*}\quad(6\text{-}3)$$

$$\{(S_2(y,f_2(x,y)),f(x,y))\}_{x,y\in\{0,1\}^*}\stackrel{c}{\equiv}\{(\text{View}_2^{\Pi}(x,y),\ \text{Output}^{\Pi}(x,y))\}_{x,y\in\{0,1\}^*}\quad(6\text{-}4)$$

这里 $\text{View}_1^\Pi(x,y)$、$\text{View}_2^\Pi(x,y)$、$\text{Output}_1^\Pi(x,y)$ 和 $\text{Output}_2^\Pi(x,y)$ 是相关的随机变量，而随机变量 $\text{Output}_i^\Pi(x,y)$ 由 $\text{View}_i^\Pi(x,y)$ 完全确定。证明协议的安全性时对于这些随机变量之间相关性保持的证明至关重要。

对于确定性功能函数情形，等式(6-1)说明，每个参与方的视图仅仅根据输入和输出就可以模拟出来。因为协议运行过程中每个参与方收到的消息都包含在视图中，这说明参与方通过协议交互所得蕴涵于他自己的输出当中，也就是说，协议的交互过程(除了输出中蕴涵的信息之外)没有泄露更多的信息，因而协议是安全的。另外，注意到等式(6-1)与等式(6-3)应用于确定性函数时相同，因为当功能函数是确定性函数时，对于每个输入对 (x,y)，必然有 $\text{Output}^\Pi(x,y)=f(x,y)$。

相对于确定性函数，在等式(6-3)和式(6-4)中，考虑协议计算随机的功能函数时，增加了 $\text{Output}^\Pi(x,y)$。此时协议 Π 计算的是随机功能函数，等式 $\text{Output}^\Pi(x,y)=f(x,y)$ 未必成立，因为等式的两边不再是具体的数值，而是两个随机变量。实际上，这两个随机变量要求分布相等，但是分布相等并不能保证式(6-1)能够推出式(6-3)，也就是说，对于随机功能函数来说，仅满足等式(6-1)不能保证协议可保密计算功能函数，下面举一个反例。

例 6.1 功能函数 $f:(1^n,1^n)\mapsto(r,\lambda)$，其中 r 在 $\{0,1\}^n$ 上均匀分布。其中 1^n 是安全参数，λ 表示空串。构造两方协议 Π_1：P_1(第一方)均匀选取 $r\in\{0,1\}^n$，将 r 发送给 P_2(第二方)，P_1 输出 r，P_2 不输出。显然，协议 Π_1 可计算功能函数 f，但是不能保密计算 f，这是因为根据功能函数 f，P_2 不知道 P_1 的输出，而协议 Π_1 中，P_2 获知 P_1 的输出。可以构造出模拟算法 $S_2(1^n)$，$S_2(1^n)$ 均匀选择 $r\in\{0,1\}^n$ 并且输出 r，易见 $S_2(1^n)$ 满足等式(6-1)但是不能满足等式(6-3)。关键点在于，等式(6-3)和式(6-4)刻画了一个参与方的输出与对方视图之间的关系，并且要求这种关系也要被模拟算法保持，即模拟算法不仅要模拟自身的视图，也要将自身视图与对方输出的关系模拟出来，因为协议是个交互过程，将一方的视图与对方的输出割裂开来，而不是作为一个整体考虑，从安全性角度来看是不完备的。

下面定义 6.2 是半诚实模型中的安全两方计算的另外一种定义方式，采用实际协议/理想模型这样的基本框架。理想模型是由两个参与方和可信第三方组成，计算由可信第三方完成。一个协议关于某种特定敌手行为称为是安全的，如果这种敌手攻击实际协议所得可以通过攻击相应的理想模型所模拟。这里模拟的概念指的是对两个参与方的联合视图的模拟。

定义 6.2(半诚实模型中的安全性) 设 $f:\{0,1\}^*\times\{0,1\}^*\rightarrow\{0,1\}^*\times\{0,1\}^*$ 是一个功能函数，$f_1(x,y)$ 和 $f_2(x,y)$ 分别是 $f(x,y)$ 的第一个和第二个分量。Π 是计算 f 的两方协议。令 $\overline{B}=(B_1,B_2)$ 是一对概率多项式时间算法，表示理想模型中两个参与方采用的算法。算法对 $\overline{B}=(B_1,B_2)$ 称为可容许的，如果至少存在一个 B_i，使得 $B_i(u,v,z)=v$，其中 u 表示 B_i 的本地输入，v 表示 B_i 的本地输出，z 表示 B_i 的辅助输入。理想模型中 f 关于 $\overline{B}=(B_1,B_2)$ 的联合执行记为 $\text{Ideal}_{f,\overline{B}(z)}(x,y)$，是以下的三元组 $(f(x,y),B_1(x,f_1(x,y),z),$ $B_2(y,f_2(x,y),z))$(理想模型中至少存在一个诚实方，将可信方发送来的输出直接作为输出)。令 $\overline{A}=(A_1,A_2)$ 是一对概率多项式时间算法，表示实际协议中两个参与方采用的算法。算法对 $\overline{A}=(A_1,A_2)$ 称为可容许的，如果至少存在一个 i，对每一个 view 和 aux，有 $A_i(\text{view},\text{aux})=\text{out}$，其中out 表示视图 view 中蕴涵的输出。实际协议中 Π 关于 $\overline{A}=(A_1,A_2)$ 的联合执行记为 $\text{Real}_{\Pi,\overline{A}(z)}(x,y)$，是以下的三元组 $(\text{Output}^\Pi(x,y),A_1(\text{View}_1^\Pi(x,y),z),$

$A_2(\mathrm{View}_2^{\Pi}(x,y),z)$)。协议 Π 称为在半诚实模型中安全计算功能函数 f，如果对于实际协议中每对可容许概率多项式时间算法 $\overline{A}=(A_1,A_2)$，都存在理想模型中可容许算法对 $\overline{B}=(B_1,B_2)$，使得

$$\{\mathrm{Ideal}_{f,\overline{B}(z)}(x,y)\}_{x,y,z} \overset{c}{\equiv} \{\mathrm{Real}_{\Pi,\overline{A}(z)}(x,y)\}_{x,y,z}$$

其中 $x,y,z\in\{0,1\}^*$，满足 $|x|=|y|$ 且 $|z|=\mathrm{poly}(|x|)$。

容易证明上述两个定义是等价的，具体证明过程请参阅文献[1]。由于这两个定义的等价性，如果所要计算的功能函数是确定性的，则采用形式简单的定义 6.1。

6.1.2 恶意模型中的安全两方计算

恶意模型是指敌手的攻击行为是恶意的，可以完全不遵守协议指令运行。对于恶意敌手，有以下 3 种情形无论采用何种协议都不可避免。

(1) 参与方拒绝参与协议运行。

(2) 参与方以替换过的输入参与协议运行。

(3) 参与方中断协议运行。

既然上述 3 种行为不可避免，那么若敌手只能施行这 3 种攻击之一(其他的攻击不能成功)就可以认为这样的协议是安全的。按照实际协议/理想模型(也称现实模型/理想模型)的定义方式，关于恶意敌手的理想模型中要对上述 3 种行为做相应的约定。

关于恶意敌手的理想模型。实际协议中不可避免的 3 种恶意行为，相应地在理想模型中要允许出现，也就是说，即使有可信第三方的存在，也不能避免某些恶意行为的发生。具体地讲，理想模型允许参与方不参与协议运行，允许参与方替换输入，显然可信方不能阻止这两种行为。另外，赋予第一方"叫停"可信方的权利，即第一方收到自己的输入之后，在可信方发送输出给第二方之前叫停可信方，这样，第一方获得输出，而第二方没有得到输出。

定义 6.3(关于恶意敌手的理想模型) 设 $f:\{0,1\}^*\times\{0,1\}^*\rightarrow\{0,1\}^*\times\{0,1\}^*$ 是一个功能函数，$f_1(x,y)$ 和 $f_2(x,y)$ 分别是 $f(x,y)$ 的第一个和第二个分量。令 $\overline{B}=(B_1,B_2)$ 是一对概率多项式时间算法，表示理想模型中两个参与方采用的算法。算法对 $\overline{B}=(B_1,B_2)$ 称为可容许的，如果至少存在一个诚实方 B_i，使得 $B_i(u,z,r)=u$，$B_i(u,z,r,v)=v$，其中 u 表示 B_i 的本地输入，v 表示 B_i 的本地输出，z 表示 B_i 的辅助输入。理想模型中 f 关于 $\overline{B}=(B_1,B_2)$ 的联合执行记为 $\mathrm{Ideal}_{f,\overline{B}(z)}(x,y)$，有以下定义：

(1) 第一方是诚实方，则 $\mathrm{Ideal}_{f,\overline{B}(z)}(x,y)$ 为 $(f_1(x,y'),B_2(y,z,r,f_2(x,y')))$，其中 $y'=B_2(y,z,r)$；

(2) 第二方是诚实方，则 $\mathrm{Ideal}_{f,\overline{B}(z)}(x,y)$ 为

$$\begin{cases}(B_1(x,z,r,f_1(x',y),\perp),\perp) & \text{若}(B_1(x,z,r,f_1(x',y)))=\perp\\ (B_1(x,z,r,f_1(x',y)),f_2(x',y)) & \text{若}(B_1(x,z,r,f_1(x',y)))\neq\perp\end{cases}$$

其中 $x'=B_1(x,z,r)$。

定义 6.4(关于恶意敌手的实际协议) 设 f 是定义 6.3 中的功能函数，Π 是计算 f 的两方协议。令 $\overline{A}=(A_1,A_2)$ 是一对概率多项式时间算法，表示实际协议中两个参与方采用的算法。算法对 $\overline{A}=(A_1,A_2)$ 称为可容许的，如果至少存在一个诚实方 A_i。实际协议中 Π 关于 $\overline{A}=(A_1,A_2)$ 的联合执行记为 $\mathrm{Real}_{\Pi,\overline{A}(z)}(x,y)$，定义为根据 $A_1(x,z)$ 和 $A_2(y,z)$ 交互产生的输出对。

定义 6.5(恶意模型中的安全性) 设功能函数 f 和协议 Π 如定义 6.4,协议 Π 称为在恶意模型中安全计算功能函数 f,如果对于实际协议中每对可容许概率多项式时间算法 $\overline{A}=(A_1,A_2)$,都存在理想模型中可容许算法对 $\overline{B}=(B_1,B_2)$,使得

$$\{\mathrm{Ideal}_{f,\overline{B}(z)}(x,y)\}_{x,y,z} \stackrel{c}{\equiv} \{\mathrm{Real}_{\Pi,\overline{A}(z)}(x,y)\}_{x,y,z}$$

其中,$x,y,z\in\{0,1\}^*$,满足 $|x|=|y|$ 且 $|z|=\mathrm{poly}(|x|)$。

定义 6.5 蕴涵了一些重要性质,如对于恶意敌手的保密性和对于诚实参与方的正确性。其中对于恶意敌手的保密性是指敌手通过与诚实方的交互所得都可以通过其局部输出推导得出,这样对恶意敌手而言,协议的交互过程并未提供额外信息。对于诚实方的正确性是指诚实方得到的输出结果与其提供的输入相符,而恶意敌手提供的输入与诚实方的输入无关。

6.2 两方保密计算功能函数

安全两方计算的最终目标是设计一般性的安全协议,使之能够抵抗任意可行的敌手攻击。要完成这个最终目标,需要分成两个步骤实施。本节完成第一步,设计保密计算任意功能函数的协议,即协议对于半诚实敌手是安全的。在 6.3 节把抵抗半诚实敌手的协议转化为抵抗恶意敌手攻击的协议。

设计抵抗任意半诚实敌手攻击的协议的基本思路如下:首先将要完成的理想功能函数表示为布尔电路,然后将这个布尔电路转化成一个协议,称之为电路赋值协议。具体转化过程是:协议由电路的输入线开始,电路的每个门作为协议的一个基本步骤,每个基本步骤中,两个参与方分别持有输入线的分享值,这一步结束后,他们分别持有输出线的分享值。每个参与方所持有的分享值得不到关于对应值的任何信息,但是将两个分享值组合即可重构出对应值。这样将保密计算电路归约到保密计算电路中的每个门,每个门的输入线由两方分享。归约是本节的一个中心概念,见定义 6.6。将由功能函数 g 到 f 的归约和保密计算函数 f 的协议复合,可以得到保密计算函数 g 的协议。这样,可以将保密计算一般功能函数归约为保密计算确定性功能函数。对于每个确定性功能函数,可设计电路赋值协议来完成它,电路赋值协议可以归约为与门和异或门的计算,为记号方便,用 GF(2)上的算术电路代替布尔电路,这样布尔电路的与门对应 GF(2)上的乘法门,而布尔电路的异或门对应 GF(2)上的加法门。乘法门的计算可归约为健忘传输协议。对于任意功能函数,如果存在安全的健忘传输协议,则根据归约定理(定理 6.1)就能够构造出计算该功能函数的协议。

6.2.1 半诚实模型中的复合定理

下面介绍归约(即保密归约)的概念及归约定理(即半诚实模型中的复合定理)。将保密计算一个功能函数归约为保密计算另一个功能函数,与通常意义下的归约概念基本类似。通常意义下的归约是借助预言器(Oracle)定义的,这里关于协议的归约也是利用了预言器。此时的预言器被两个参与方调用,每个参与方向预言器提交询问,预言器将答案返回给相应的参与方。

定义 6.6(保密归约) 一个预言器辅助协议称为应用预言函数 f,如果预言器按照函数 f 回答询问。即当预言器被调用,第一方提交的询问是 q_1,第二方提交的询问是 q_2,则预言器的回答是 $f(q_1,q_2)$。一个应用预言函数 f 的预言器辅助协议称为保密计算功能函数 g,

如果存在多项式时间算法 S_1 和 S_2 分别满足式(6.3)和式(6.4),一个预言器辅助协议称为保密归约 g 到 f,如果此协议应用预言函数 f 时则保密计算功能函数 g。

定理 6.1(半诚实模型中的复合定理) 设功能函数 g 保密归约到 f,并且存在协议保密计算功能函数 f,那么存在一个协议保密计算功能函数 g。

限于篇幅,这里就不给出定理的具体证明过程,感兴趣的读者可参阅文献[1]。

给定一个一般的功能函数 g,利用下面介绍的预言辅助协议 6.1,可以将其归约到某个确定性功能函数 f。首先,令 $g(r,(x,y))$ 表示选择随机串 r 时 $g(x,y)$ 的取值。定义确定性功能函数 f:$f((x_1,r_1),(x_2,r_2))=g(r_1\oplus r_2,(x_1,x_2))$。

协议 6.1

输入:第一方的输入是 $x_1\in\{0,1\}^n$,第二方的输入是 $x_2\in\{0,1\}^n$。

(1) 第一方均匀选取随机串 $r_1\in\{0,1\}^{\text{poly}(|x_1|)}$,第二方均匀选取随机串 $r_2\in\{0,1\}^{\text{poly}(|x_2|)}$。

(2) (归约)第一方和第二方分别以询问 (x_1,r_1) 和 (x_2,r_2) 调用预言器,并且记录预言器的回答。

输出:每个参与方将预言器的回答作为输出。

易证,协议 6.1 保密计算功能函数 g,即协议 6.1 将功能函数 g 保密归约到 f。

6.2.2 半诚实模型中安全的健忘传输协议

健忘传输协议在安全多方计算中有着重要的作用。令 k 是一个固定的正整数,$\sigma_1,\cdots,\sigma_k\in\{0,1\}$,$i\in\{1,\cdots,k\}$。健忘传输协议要完成的功能函数记为 OT_1^k,定义如下:$\text{OT}_1^k((\sigma_1,\sigma_2,\cdots,\sigma_k),i)=(\lambda,\sigma_i)$。习惯上将第一方称为发送方,持有输入 $(\sigma_1,\sigma_2,\cdots,\sigma_k)$,第二方称为接收方,持有输入 i。功能函数 OT_1^k 要完成的功能或者目标是将发送方的第 i 个比特传输给接收方,接收方不能获知其他位置的比特,即不能获知 $\sigma_j,j\neq i$,发送方也不能知道接收方要求收到哪个位置的比特,即发送方不能知道 i。本书第 11 章专门介绍了健忘传输协议方面的研究成果,但是为了本章的需要首先简要介绍一点健忘传输协议的基本知识。建议读者在阅读本节时可事先阅读一下本书的11.1 节。

定义 6.7(加强陷门置换族) 设$\{f_\alpha:D_\alpha\to D_\alpha\}$是一个陷门置换族,其上定义 4 个算法,分别是指标算法 I、抽样算法 D、求值算法 F 和求逆算法 B。给定输入 1^n,算法 I 从置换族中选择一个置换 f_α 的下标 α 以及相应的陷门 τ;给定输入 α,算法 D 从置换 f_α 的定义域中抽样,输出一个在定义域中均匀分布的 x;给定输入 α 和 x,算法F 返回 $f_\alpha(x)$;给定 f_α 的值域中的 y 及(α,τ),算法 B 返回 $f_\alpha^{-1}(y)$。称一个陷门置换族为加强陷门置换族,如果对任意概率多项式时间算法 A,任意正多项式 p,所有充分大的 n,有

$$\Pr[A(I_1(1^n),R_n)=f^{-1}_{I_1(1^n)}(D'(I_1(1^n),R_n))]<\frac{1}{p(n)}$$

其中 $I_1(1^n)$表示算法 I 输出中的第一个分量即下标,D'是执行算法 D 的两输入算法,将 D 的掷币结果作为辅助输入提供给 D'。

假设加强陷门置换族存在,则可以设计保密计算健忘传输功能函数 OT_1^k 的安全协议。设$\{f_\alpha:D_\alpha\to D_\alpha\}_{\alpha\in I}$是加强陷门置换族,$b$ 是这一族陷门置换的硬核谓词(Hard-core Predicate)。简单地讲,一个多项式可计算的谓词 $b:\{0,1\}^*\to\{0,1\}$被称为是一个函数 f 的

硬核谓词，如果对每一个有效的算法，在给定 $f(x)$ 的情况下，能以略大于 1/2 的成功概率猜中 $b(x)$。在以下的协议 6.2 的描述中，将发送方简记为 S，接收方简记为 R。该协议的安全性依赖于辅助的安全参数 1^n，随着 n 的不断增大，该协议的安全性程度增高。大家知道，协议的安全性定义为实际执行过程中的视图与理想模型中执行过程这样两个随机变量的计算不可区分性，也就是说，随着 n 的增大，这两个随机变量的接近程度越来越好。

协议 6.2（基于加强陷门置换族的健忘传输协议）

输入：S 的输入是 $(\sigma_1,\sigma_2,\cdots,\sigma_k)\in\{0,1\}^k$，$R$ 的输入是 $i\in\{1,2,\cdots,k\}$，S 和 R 的辅助输入是安全参数 1^n。

(1) S 均匀选择随机串 r，利用指标-陷门对生成算法 G，生成一对指标-陷门对 $(\alpha,t)=G(1^n,r)$。S 发送指标 α 给 R。

(2) 首先 R 在加强陷门置换族 $\{f_\alpha:D_\alpha\to D_\alpha\}_{\alpha\in I}$ 的定义域 D_α 中均匀且独立地选取随机串 $r_1,\cdots,r_k$，调用 k 次定义域抽样算法，产生定义域中 k 个数，即 $x_j=D(\alpha,r_j)$，其中 $j=1,\cdots,k$。其次 R 计算 $y_i=f_\alpha(x_i)$，对于 $j\neq i$，令 $y_j=x_j$。最后 R 发送 $(y_1,y_2,\cdots,y_k)$ 给 S。

(3) S 收到 $(y_1,y_2,\cdots,y_k)$ 之后，利用陷门 t 计算 $z_j=f_\alpha^{-1}(y_j)$，$j=1,\cdots,k$。S 发送 $(c_1,c_2,\cdots,c_k)=(\sigma_1\oplus b(z_1),\sigma_2\oplus b(z_2),\cdots,\sigma_k\oplus b(z_k))$ 给 R。

(4) R 接收到 S 在第(3)步发送来的消息 $(c_1,c_2,\cdots,c_k)$，计算 $c_i\oplus b(x_i)$ 并将结果输出。

易知，协议 6.2 计算功能函数 OT_1^k。下面结论说明协议 6.2 保密计算功能函数 OT_1^k。

命题 6.1 设 $\{f_i:D_i\to D_i\}$ 是加强陷门置换族，b 是此陷门置换族的硬核谓词。那么协议 6.2 在半诚实模型中保密计算功能函数 OT_1^k。

证明 由于 OT_1^k 是确定性功能函数，则可以应用定义 6.1 中的简单形式。欲证明协议 6.2 保密计算 OT_1^k，需对两方的视图分别设计模拟器算法，使得模拟器算法只给定输入和输出即可产生出相应的视图。

发送方的视图由输入、随机串、接收到的消息组成，即 $\mathrm{View}_1^\Pi(x,y)=\{((\sigma_1,\cdots,\sigma_k),1^n),r,(y_1,\cdots,y_k)\}$，其中 $((\sigma_1,\cdots,\sigma_k),1^n)$ 是提供给发送方的输入，r 是发送方在第(1)步选取的随机串，$(y_1,\cdots,y_k)$ 是发送方在第(3)步接收到的消息。首先对于发送方的视图设计模拟器算法 S_1，提交给模拟器 S_1 的输入是 $(((\sigma_1,\cdots,\sigma_k),1^n),\lambda)$，其中 $((\sigma_1,\cdots,\sigma_k),1^n)$ 是第一方的输入，λ 是第一方的输出，S_1 的构造如下：S_1 均匀选择随机串 r' 作为被模拟的发送方的随机串，S_1 独立且均匀地生成 $y_1',\cdots,y_k'\in D_\alpha$。$S_1$ 输出 $\{((\sigma_1,\cdots,\sigma_k),1^n),r',(y_1',\cdots,y_k')\}$。显然，这样构造的算法 S_1 的输出分布与实际协议中发送方的视图的分布相等。

接收方的视图 $\mathrm{View}_2^\Pi(x,y)=\{(i,1^n),\alpha,(r_1,\cdots,r_k),(c_1,\cdots,c_k)\}$，其中 $(i,1^n)$ 是接收方的输入，α 是第(1)步收到的消息，$(r_1,\cdots,r_k)$ 是接收方选择的随机串，$(c_1,\cdots,c_k)$ 是第(3)步接收到的消息。构造接收方的模拟器算法 S_2 如下：S_2 的输入是 $((i,1^n),\sigma)$，其中 $(i,1^n)$ 是接收方的输入，σ 是接收方的输出。S_2 均匀选择随机数 r^{S_2}，运行生成算法得到指标-陷门对 $(\alpha',t')=G(1^n,r^{S_2})$。$S_2$ 均匀且独立地选择 $r_1',\cdots,r_k'$，计算 $x_j'=D(\alpha',r_j')$，$j=1,\cdots,k$，令 $y_i'=f_{\alpha'}(x_i')$，对于 $j\neq i$，令 $y_j'=x_j'$，置 $c_i'=\sigma_i\oplus b(x_i')$，对于 $j\neq i$，均匀选取 $c_j'\in\{0,1\}$。S_2 输出 $\{(i,1^n),\alpha',(r_1',\cdots,r_k'),(c_1',\cdots,c_k')\}$。显然，$S_2$ 输出中的前 3 个分量与实际协议接收方的视图的相应部分分布相等，进一步，即使加上 c_i，这两个分布也是相等的，那么这两个随机变量只在不等于 i 的 c_j 和 c_j' 处可能分布不同，在实际协议中，$c_j=\sigma_j\oplus b(f_\alpha^{-1}(y_j))=\sigma_j\oplus b(f_\alpha^{-1}(x_j))$，而在模拟器算法中，$c_j'$ 是 $\{0,1\}$ 上均匀分布的比特。然而，根据加强陷门置换族的性质只给

定 α 和 r_j，不能求逆，这样判断 $b(f_\alpha^{-1}(x_j))$ 是 0 或 1 没有优势，因此这两个分布是计算不可区分的。

6.2.3 保密计算 $c_1+c_2=(a_1+a_2)\cdot(b_1+b_2)$

本小节说明乘法门(乘法函数)可以归约为功能函数 OT_1^4。乘法函数是指两个参与方计算功能函数 $((a_1,b_1),(a_2,b_2))\mapsto(c_1,c_2)$，其中 a_1+a_2 是第一方的输入，b_1+b_2 是第二方的输入，满足 $c_1+c_2=(a_1+a_2)\cdot(b_1+b_2)$。协议 6.3 将乘法功能函数保密归约到功能函数 OT_1^4，注意这里为了简便，讨论的都是在 GF(2)上的运算。

协议 6.3

输入：第一方输入是 $(a_1,b_1)\in\{0,1\}^2$，第二方输入是 $(a_2,b_2)\in\{0,1\}^2$。

(1) 第一方均匀选取 $c_1\in\{0,1\}$。

(2) 两方联合调用功能函数 OT_1^4。第一方以发送方的身份调用，第二方以接收方的身份调用。第一方利用输入对 (a_1,b_1) 以及第(1)步选定的 c_1，计算四元组 $(((a_1+0)\cdot(b_1+0)+c_1),((a_1+0)\cdot(b_1+1)+c_1),((a_1+1)\cdot(b_1+0)+c_1),((a_1+1)\cdot(b_1+1)+c_1))$，以此四元组作为调用功能函数 OT_1^4 的输入。第二方利用输入对 (a_2,b_2)，计算 $1+2a_2+b_2\in\{1,2,3,4\}$ 作为调用功能函数 OT_1^4 的输入。这样在调用功能函数 OT_1^4 结束后，第一方从功能函数 OT_1^4 处得到空串 λ，第二方得到 $(1+2a_2+b_2)$ 位置的运算结果，即 $(a_1+a_2)\cdot(b_1+b_2)+c_1$。

输出：第一方输出 c_1，第二方输出从功能函数 OT_1^4 处得到的结果。

显然，协议 6.3 计算乘法功能函数，另外关于两个参与方的模拟器算法也容易构造，因为协议 6.3 实际上是没有交互的协议。

6.2.4 电路赋值协议

本节证明计算任意表示成 GF(2)上的算术电路的确定性功能函数，能够保密归约到计算乘法函数。首先将要计算的功能函数表示成电路，电路的输入线共 $2n$ 条，每个参与方有 n 条输入线，为简便起见，假设每个参与方有 n 个输出比特。

协议 6.4(将电路赋值函数归约到乘法功能函数)

输入：第一方持有输入 $(x_1^1,\cdots,x_1^n)\in\{0,1\}^n$，第二方持有输入 $(x_2^1,\cdots,x_2^n)\in\{0,1\}^n$。

(1) 分享输入。每个参与方将输入的每个比特与对方分享。第一方均匀选择一列比特串 $r_1^1,\cdots,r_1^n$ 发送给第二方，这样，第一方的每个输入线分成两部分，第一方自己持有输入线的分享是 $x_1^1+r_1^1,\cdots,x_1^n+r_1^n$，第二方持有的第一方输入线的分享是 $r_1^1,\cdots,r_1^n$。第二方用同样的方法将自己的输入线分享，第一方持有的分享是 $r_2^1,\cdots,r_2^n$，第二方自己持有的分享是 $x_2^1+r_2^1,\cdots,x_2^n+r_2^n$。

(2) 电路赋值。根据电路的线路顺序，对于电路中的每个门有两条输入线路，两个参与方利用关于这两条输入线路的各自的分享值，保密计算门的输出的分享。两个参与方分别持有某个门两条输入线的分享，即第一方持有分享值 a_1,b_1，第二方持有分享值 a_2,b_2，其中 a_1,a_2 是第一条输入线的分享，即 a_1+a_2 是第一条输入线上的输入，b_1,b_2 是第二条输入线的分享，即 b_1+b_2 是第二条输入线上的输入。因为讨论 GF(2)上的算术电路，因此只需设计协议保密计算加法门和乘法门两种具体的门运算。

加法门赋值：第一方持有的关于两条输入线的分享分别是 a_1, b_1，第二方持有的关于两条输入线的分享分别是 a_2, b_2。完成加法门运算的协议很平凡，第一方将加法门的输出线的分享设置为 a_1+b_1，第二方将此门的输出线的分享设置为 a_2+b_2，即两方将自己的输入分享值分别相加得到输出的分享值。

乘法门赋值：两个参与方以各自关于输入的分享值 (a_1, b_1) 和 (a_2, b_2) 调用乘法功能函数，以函数返回的回答 c_1、c_2 作为乘法门输出的各自的分享值。根据乘法功能函数，两方的输出 c_1、c_2 满足 $c_1+c_2=(a_1+b_1)\cdot(a_2+b_2)$。

(3) 恢复输出。一旦整个电路的输出线的分享确定，则每个参与方将每条输出线的分享值发送到对方相应的输出线，将每条输出线上获得的计算结果的分享值与从对方收到的分享值相加，即确定出每条输出线上的比特。

输出：将输出线上的比特输出。

下面结论说明协议 6.4 将计算某个电路功能函数归约为乘法功能函数，并且归约是保密的。

命题 6.2　协议 6.4 将电路赋值功能函数保密归约为乘法功能函数。

证明　由于电路赋值功能函数是确定性功能函数，因此采用定义 6.1 中的简单形式，即只要构造模拟器算法，模拟出每个参与方自身的视图即可。因为两个参与方的对称性，不失一般性，对第一方的视图提出模拟器算法，第二方视图模拟器算法的构造类似。设第一方的输入是 $x_1^1, \cdots, x_1^n$，输出是 $y_1^1, \cdots, y_1^n$，视图是 $\{(x_1^1, \cdots, x_1^n), (r_1^1, \cdots, r_1^n), V^1, V^2, V^3\}$，其中 $V^1=(r_2^1, \cdots, r_2^n)$ 是第(1)步中从第二方接收到的消息(即第二方输入的分享)，V^2 是从乘法函数预言器收到的消息，V_3 是在第(3)步中从第二方接收到的关于输出的分享。模拟器算法 S_1 构造如下：S_1 的输入是 $\{(x_1^1, \cdots, x_1^n), (y_1^1, \cdots, y_1^n)\}$。$S_1$ 均匀选取 $r_1^1, \cdots, r_1^n$ 和 $r_2^1, \cdots, r_2^n$，其中 $r_1^1, \cdots, r_1^n$ 是第一方的随机数，$r_2^1, \cdots, r_2^n$ 模拟第(1)步中从第二方收到的消息，S_1 将第 j 条输入线的分享设置为 $x_1^j+r_1^j$，将关于第二方输入线的分享设置为 $r_2^1, \cdots, r_2^n$。对于电路计算，S_1 逐步地按照所给电路计算，将加法门输出线的分享设置为输入线的分享值的和，乘法门输出线的分享设置为从 $\{0,1\}$ 中均匀选取的一个比特。对于实际协议视图中的 V_3 消息，S_1 的设置方法是：将对应输出线的分享值 w_j 与输出 y_j 相加，即根据已有的输出确定消息 V_3，注意 V_3 在实际协议中是来自于第二方的消息而模拟器算法必须由 S_1 自己产生。这样在模拟器算法中能够保持与实际协议中一致的 V_3 与输出之间的关系。S_1 输出 $\{(x_1^1, \cdots, x_1^n), (r_1^1, \cdots, r_1^n), V^1, V^2, V^3\}$，易见，$S_1$ 输出的随机变量与实际协议视图的随机变量是同分布的，从而命题得证。

根据前面一系列归约，即一般功能函数可以归约到确定性功能函数，确定性功能函数可以归约到乘法门计算函数，乘法门计算可以归约到 OT_1^4，这样可以将任意功能函数归约到功能函数 OT_1^4，而如果加强陷门置换族存在，则 OT_1^4 能够被保密计算。因此对于半诚实模型，有以下的基本定理。

定理 6.2　假设存在加强陷门置换族，则任意功能函数在半诚实模型中都可以保密计算。

6.3　安全两方计算的基本定理

本节的目的是把抵抗半诚实敌手的协议转化为抵抗恶意敌手攻击的协议，建立安全两方计算的主要结果(定理 6.3)。

定理 6.3(基本定理) 假设存在加强陷门置换族,则任意功能函数在恶意模型中都可以安全计算。

定理 6.3 的建立是将半诚实模型中的任意一个协议编译成恶意模型中的协议。编译器的工作原理是强制参与方或者以半诚实行为执行协议或者被检测出欺骗,对于后者,协议中断执行。

6.3.1 恶意模型中的复合定理

首先明确恶意参与方可能的恶意行为(超出半诚实参与方的行为)。一个恶意参与方可能替换输入,即没有以给定的输入开始执行协议,输入替换是不可避免的,对于这种恶意行为,编译器要保证的是这种替换与对方的输入无关,只依赖于自身的原始输入。经过编译的协议有输入承诺阶段,就是为了达到这一目的。输入承诺阶段应用的工具是承诺方案和零知识证明系统。承诺方案是两阶段的双方协议,两个阶段分别是承诺阶段和公开阶段。承诺方持有秘密比特 b,在承诺阶段,承诺方发送关于 b 的一个比特串 C_b 给接收方,要求此时接收方根据 C_b 不能获得关于 b 的信息,这个性质称为隐蔽性(或隐藏性);在公开阶段,承诺方将 C_b 公开,要求此时承诺方不能将 C_b 公开为另一个 b',这个性质称为约束性(或绑定性)。一个恶意参与方可能不按照均匀分布选择随机串,对于这种情况,编译器要强制恶意参与方应用均匀分布的随机带。经过编译的协议中有掷币生成阶段,就是为了达到这一目的。掷币生成阶段应用的工具是扩展的掷币入井协议。一个恶意参与方还可能不按照协议指令发送消息,需要强制参与方发送的消息与已经承诺过的输入和随机带相符。编译器的协议模仿阶段就是为达到这一目标,应用的工具是零知识证明系统。

这样,由原始协议出发,经过编译器编译之后的协议由以下 3 部分组成。

(1) 输入承诺阶段。每个参与方应用输入承诺功能函数对其输入作承诺。此处输入承诺功能函数保证承诺方确实知道他所承诺的值,并且保持被承诺值的秘密性。

(2) 掷币生成阶段。参与协议双方利用掷币功能函数生成随机带。每个参与方获得均匀分布的一个随机带,对方获得此随机带的承诺值。

(3) 协议模仿阶段。参与协议双方利用认证计算功能函数完成对原始协议每一步的模仿执行。认证计算功能函数保证或者协议被正确执行,或者被检测出欺骗从而中断。

类似于半诚实模型,对于恶意模型也是首先给出归约的概念。因为处理的是恶意模型,所以相对于半诚实模型的保密归约,这里讨论的是安全归约。

类似于保密归约,一个预言器辅助协议称为应用预言函数 f,如果预言器按照函数 f 回答询问,即当预言器被调用,第一方提交的询问是 q_1,第二方提交的询问是 q_2,则预言器的回答是 $f(q_1,q_2)$。

令 $\overline{A}=(A_1,A_2)$是一对概率多项式时间算法,表示预言器辅助协议中两个参与方采用的算法。算法对 $\overline{A}=(A_1,A_2)$称为可容许的,如果至少存在一个 i 诚实执行预言器辅助协议。一个应用预言函数 f 的预言器辅助协议中关于 $\overline{A}=(A_1,A_2)$的联合执行记为 $\mathrm{Real}^f_{\Pi,\overline{A}(z)}(x,y)$,是算法 $A_1(x,z)$和 $A_2(y,z)$交互产生的输出对,其中调用预言器的回答是依据函数 f。一个应用预言函数 f 的预言器辅助协议 Π 称为安全计算功能函数 g,如果对于预言器辅助的实际协议 Π 的任意概率多项式时间可容许算法对 $\overline{A}=(A_1,A_2)$,存在理想模型中的概率多项式时间可容许算法对 $\overline{B}=(B_1,B_2)$,使得 $\{\mathrm{Ideal}_{g,\overline{B}(z)}(x,y)\}_{x,y,z}\overset{c}{\equiv}\{\mathrm{Real}^f_{\Pi,\overline{A}(z)}(x,y)\}_{x,y,z}$。一个

预言器辅助协议称为安全归约 g 到 f，如果此协议应用预言函数 f 时则安全计算功能函数 g。

定理 6.4（恶意模型中的复合定理） 设功能函数 g 安全归约到 f，并且存在协议安全计算功能函数 f，那么存在一个协议安全计算功能函数 g。

限于篇幅，这里就不给出定理 6.4 的详细证明，感兴趣的读者可参阅文献[1]中的定理 7.13.3。应用与 6.2.1 小节同样的方法可将随机功能函数归约到确定性功能函数。即设 g 是一个随机功能函数，定义 $f((x_1,r_1),(x_2,r_2))=g(r_1\oplus r_2,(x_1,x_2))$，其中 $g(r,(x,y))$ 表示选择的随机串是 r 时 $g(x,y)$ 的取值，则协议 6.1 将随机功能函数 g 归约到确定性功能函数 f。

6.3.2　编译器调用的功能函数

6.3.1 小节指出，经由编译器产生的安全协议需要调用 3 个功能函数，分别是输入承诺、扩展掷币及认证计算功能函数。这里给出这些功能函数的定义以及安全计算它们的具体协议，上述 3 个功能函数的基础是掷币功能函数、限制认证计算功能函数及像传输功能函数。

首先提出掷币功能函数的定义并设计掷币入井协议安全计算此功能函数。掷币功能函数：$(1^n,1^n)\mapsto(b,b)$，其中 b 在 $\{0,1\}$ 上均匀分布。掷币功能函数刻画互不信任的双方得到一个均匀分布的比特这样的功能。安全计算此功能函数的协议称为掷币入井协议，即协议 6.5。

协议 6.5（掷币入井协议） 对任意 r，令 $C_r:\{0,1\}\to\{0,1\}^*$。

输入：双方的输入是安全参数 1^n。

(1) 第一方均匀选择 $\sigma\in\{0,1\}$ 和 $s\in\{0,1\}^n$，并且发送 $c=C_s(\sigma)$ 给第二方。

(2) 第二方均匀选择 $\sigma'\in\{0,1\}$，并且发送 σ' 给第一方。

(3) 第一方输出 $\sigma\oplus\sigma'$，并且发送 (σ,s) 给第二方。

(4) 第二方验证等式 $c=C_s(\sigma)$ 是否成立。如果成立，则输出 $\sigma\oplus\sigma'$，否则输出 $\perp$。

输出：第一方输出 $b=\sigma\oplus\sigma'$，第二方或者输出 b 或者输出 $\perp$。

显然，若参与协议的双方都是诚实的，则两者输出同样的均匀分布的比特。下面证明即使有一个参与方是恶意的，则或者双方输出同样的均匀分布比特或者诚实方检测出对方欺骗。

命题 6.3 设 C 是一个比特承诺方案，则协议 6.5 是掷币入井协议。

证明 要证明协议 6.5 是掷币入井协议，即协议 6.5 安全计算掷币功能函数。根据安全两方计算的定义，需要对于任意攻击实际协议的可容许对 $\bar{A}=(A_1,A_2)$，构造出攻击理想模型的可容许对 $\bar{B}=(B_1,B_2)$。所谓可容许对，是指两个参与方中至少有一方是诚实的，所采用的算法如协议规定，因此在证明过程中分两种情况讨论。

首先假设第一方是诚实的，在这种情形下，B_1 是确定的，只需构造出 B_2。B_2 将攻击实际协议的敌手算法 A_2 作为子程序调用，生成 A_2 需要的消息并且获得 A_2 发送出来的消息。根据协议 6.5，A_2 希望收到的消息是 $C_s(\sigma)$ 和 (σ,s)，A_2 发送出来的消息是 σ'。B_2 算法设计如下：B_2 将输入 1^n 发送给可信方，从可信方处获得掷币功能函数的计算结果 b，是一个均匀分布的比特。B_2 重复下述(1)、(2)两个步骤 n 次，试图生成 A_2 的视图，注意此时的视图必须以从可信方收到的 b 为输出结果，这样才能够保持视图与输出的一致性。

(1) B_2 均匀选择 $\sigma\in\{0,1\}$ 和 $s\in\{0,1\}^n$，计算 $c=C_s(\sigma)$，将 c 给 A_2，A_2 回答一个比特，记为 σ'，此时 A_2 的回答 σ' 是依赖于消息 c 的，即 $\sigma'\leftarrow A_2(c)$。

(2) 若 $\sigma\oplus\sigma'=b$，则 B_2 将 $(c,(\sigma,s))$ 给 A_2，并将算法 A_2 的输出作为 B_2 在理想模型中的输出。否则，回到第(1)步，执行下一次。如果 n 次执行 B_2 都没有成功，即每次执行都不满足 $\sigma\oplus\sigma'=b$，那么 B_2 输出一个失败记号。

B_2 输出失败记号的概率是可忽略的。这是因为对任意 $b\in\{0,1\}$，每次执行第(2)步成功的概率基本上是 1/2，第(2)步成功当且仅当 $\sigma\oplus\sigma'=b$，即 $A_2(C_s(\sigma))=b\oplus\sigma$，其中 $(\sigma,s)\in\{0,1\}\times\{0,1\}^n$ 是均匀选取的。而 $\Pr_{\sigma,s}[A_2(C_s(\sigma))=b\oplus\sigma]=\frac{1}{2}\cdot\Pr[A_2(C(0))=b]+\frac{1}{2}\cdot\Pr[A_2(C(1))=b\oplus 1]=\frac{1}{2}+\frac{1}{2}\cdot(\Pr[A_2(C(0))=b]-\Pr[A_2(C(1))=b])$，$C$ 是一个承诺方案，因此 $\Pr[A_2(C(0))=b]-\Pr[A_2(C(1))=b]$ 是 n 的可忽略函数，所以 B_2 输出失败记号的概率是可忽略的。若 B_2 没有输出失败记号，这时理想模型中的分布 $\text{Ideal}_{f,\bar{B}}(1^n,1^n)=\{b,A_2(C_s(\sigma),(\sigma,s))\}$，其中 b 是从可信方接收到的均匀分布的比特，且满足 $b=\sigma\oplus A_2(C_s(\sigma))$。相对应的实际协议的分布是 $\text{Real}_{\Pi,\bar{A}}(1^n,1^n)=\{b,A_2(C_s(\sigma),(\sigma,s))\}$，其中 b 是双方共同决定的随机比特，也满足 $b=\sigma\oplus A_2(C_s(\sigma))$。此处这两个分布有相同的形式，并且都被 (σ,s) 所确定，区别在于 $\text{Real}_{\Pi,\bar{A}}(1^n,1^n)$ 中的 (σ,s) 均匀分布在 $\{0,1\}^{n+1}$ 上，每一对出现的概率都是 $2^{-(n+1)}$，而在 $\text{Ideal}_{f,\bar{B}}(1^n,1^n)$ 中，首先确定的是比特 b，(σ,s) 均匀分布在满足 $b=\sigma\oplus A_2(C_s(\sigma))$ 对所构成的空间中，根据 b 的随机性及承诺方案的隐蔽性，易证在 $\text{Real}_{\Pi,\bar{A}}(1^n,1^n)$ 和 $\text{Ideal}_{f,\bar{B}}(1^n,1^n)$ 这两个随机变量中，各自所对应的 (σ,s) 的分布差是可忽略的，因此这两个随机变量是计算不可区分的。

其次假设第二方是诚实的，在这种情形下，B_2 是确定的，只需构造出 B_1。给定输入 1^n，B_1 将 A_1 作为子程序调用，将 A_1 期望获得的消息 $\sigma'\in\{0,1\}$ 给 A_1 并且接收 A_1 发出的消息。B_1 算法设计如下：B_1 以输入 1^n 调用 A_1，A_1 输出一个消息记为 c，c 至多是某一个比特 $\sigma\in\{0,1\}$ 的承诺值。B_1 试图获得 A_1 在第(3)步发送出的消息，因为 A_1 在第(3)步发出的消息可能依赖于第(2)步获得的消息，这样 B_1 用第(2)步所有可能的不同消息试验，以期得到 A_1 对于两种不同的第(2)步输入消息的反馈。

(1) B_1 发送 0 给 A_1 作为 A_1 第(2)步收到的消息，记录 A_1 发出的消息，或者是 $\perp$ 或者是 (σ_0,s_0)，若 $c\neq C_{s_0}(\sigma_0)$ 视同 A_1 发出 $\perp$。

(2) 将 A_1 重绕至第(2)步，B_1 发送 1 给 A_1 作为 A_1 第(2)步收到的消息，记录 A_1 发出的消息，或者是 $\perp$ 或者是 (σ_1,s_1)，若 $c\neq C_{s_1}(\sigma_1)$ 视同 A_1 发出 $\perp$。

根据 A_1 对于(1)、(2)两种情形的回答，将 B_1 的算法设计分成以下 3 种情况。

① 若 A_1 对于(1)、(2)两种情形都发出 $\perp$，那么 B_1 中断，并以 $A_1(1^n,\sigma')$ 作为输出，此处 σ' 是均匀分布的一个比特。由于此时 B_1 没有调用可信方，因此 $\text{Real}_{\Pi,\bar{A}}(1^n,1^n)=(A_1(1^n,\sigma'),\perp)=\text{Ideal}_{f,\bar{B}}(1^n,1^n)$。

② 若 A_1 只是对于(1)、(2)中某一种情形正确回答，记此比特为 σ'，定义 $\sigma=\sigma_{\sigma'}$。B_1 发送 1^n 给可信方，从可信方获得掷币功能函数的计算结果 b，如果 $b\neq\sigma\oplus\sigma'$，则中断可信方，即在理想模型中可信方不给第二方发送计算结果，如果 $b=\sigma\oplus\sigma'$，则允许可信方发送 b 给第二方。B_1 将 σ' 发送给 A_1 并以 A_1 的输出作为输出。此时，$\text{Ideal}_{f,\bar{B}}(1^n,1^n)$ 以 1/2 的概率中

断，若没有中断，则 $\text{Real}_{\Pi,\bar{A}}(1^n,1^n)=(A_1(1^n,\sigma'),\sigma\oplus\sigma')=\text{Ideal}_{f,\bar{B}}(1^n,1^n)$；中断时，$\text{Real}_{\Pi,\bar{A}}(1^n,1^n)=(A_1(1^n,\sigma'\oplus 1),\bot)=\text{Ideal}_{f,\bar{B}}(1^n,1^n)$。

③ 若 A_1 对于(1)、(2)两种情形都给出正确回答，根据承诺方案的约束性，有 $\sigma_0=\sigma_1$，定义 $\sigma=\sigma_0(=\sigma_1)$。$B_1$ 发送 1^n 给可信方，从可信方获得掷币功能函数的计算结果 b，令 $\sigma'=b\oplus\sigma$ 并允许可信方发送 b 给第二方。B_1 将 σ' 发送给 A_1 并以 A_1 的输出作为输出。$\text{Ideal}_{f,\bar{B}}(1^n,1^n)=(A_1(1^n,\sigma\oplus b),b)$，$\text{Real}_{\Pi,\bar{A}}(1^n,1^n)=(A_1(1^n,\sigma'),\sigma\oplus\sigma')$，而 σ' 和 b 是 $\{0,1\}$ 上均匀分布的比特，且 σ' 与 σ 的分布相互独立，因此 $\text{Ideal}_{f,\bar{B}}(1^n,1^n)$ 与 $\text{Real}_{\Pi,\bar{A}}(1^n,1^n)$ 的分布相等。

命题 6.3 证毕，即协议 6.5 安全计算掷币功能函数。

编译器中应用的第二个功能函数是认证计算，首先提出以下部分认证计算功能函数。令 $f:\{0,1\}^*\times\{0,1\}^*\to\{0,1\}^*$ 和 $h:\{0,1\}^*\to\{0,1\}^*$ 是多项式时间可计算的，且 h 是单射。则部分 h 认证 f 计算功能函数定义为：$(\alpha,h(\alpha))\mapsto(\lambda,f(\alpha))$。以下设计的协议 6.6 安全计算此部分认证计算功能函数。

协议 6.6（部分认证计算协议）　定义语言 $L=\{(u,v)\mid$ 存在 x，使得 $u=h(x)\wedge v=f(x)\}$，(P,V) 是语言 L 的交互证明系统。

输入：第一方输入是 $\alpha\in\{0,1\}^*$，第二方输入是 $u=h(\alpha)$。

(1) 第一方发送 $v=f(\alpha)$ 给第二方。

(2) 双方调用证明系统 (P,V)，第一方作为证明者，第二方作为验证者。证明系统的公共输入是 (u,v)，证明者的辅助输入是 α，要向验证者证明存在 x，使得 $(u=h(x))\wedge(v=f(x))$。一旦验证者拒绝证明，则第二方中断协议，输出 $\bot$。

输出：若第二方没有中断协议，即验证者接受证明，则第二方输出 v，第一方没有输出。

命题 6.4　若函数 h 是单射并且 (P,V) 是语言 L 的零知识交互证明系统，则协议 6.6 安全计算 h 认证 f 计算功能函数。

证明　根据协议的安全性定义，需要将攻击实际协议（即现实模型）的任意可容许对 (A_1,A_2)，变换到攻击理想模型的可容许对 (B_1,B_2)。分下面两种情况讨论。

(1) 第一方诚实。在这种情形下，B_1 是确定的，需要将攻击实际协议的敌手 A_2 转化为攻击理想模型的敌手 B_2，B_2 将 A_2 作为子程序调用，算法 B_2 的具体设计如下：B_2 获得输入 $u=h(\alpha)$，B_2 发送 u 给可信方并得到输出 v，调用零知识证明系统的模拟器算法，提供给模拟器的输入是 (u,v)，将 A_2 作为恶意的验证者算法，记获得的模拟过程为 $S=S(u,v)$。B_2 将合理的执行视图 (v,S) 交给 A_2，以 A_2 的输出作为输出。令 $R(\alpha)$ 表示实际协议中验证者的视图，则 $\text{Real}_{\Pi,\bar{A}}(\alpha,h(\alpha))=(\lambda,A_2(h(\alpha),f(\alpha),R(\alpha)))$，而 $\text{Ideal}_{f,\bar{B}}(\alpha,h(\alpha))=(\lambda,A_2(h(\alpha),f(\alpha),S(h(\alpha),f(\alpha))))$，根据零知识证明的定义，显然有 $\text{Real}_{\Pi,\bar{A}}(\alpha,h(\alpha))$ 与 $\text{Ideal}_{f,\bar{B}}(\alpha,h(\alpha))$ 是计算不可区分的。

(2) 第二方诚实。在这种情形下，B_2 是确定的，需要将攻击实际协议的敌手 A_1 转化为攻击理想模型的敌手 B_1，B_1 将 A_1 作为子程序调用，算法 B_1 的具体设计如下：B_1 获得输入 $\alpha\in\{0,1\}^n$，B_1 以输入 α 调用 A_1，记 A_1 发出的消息为 v，即 $v\leftarrow A_1(\alpha)$，B_1 作为诚实的验证者和 A_1 交互，此证明系统的公共输入是 $(h(\alpha),v)$，由于此证明系统有可忽略的可靠性错误 μ，因此如果 $(h(\alpha),v)\notin L$，A_1 被检测出欺骗的概率是 $1-\mu(n)$。若 B_1 拒绝 A_1 的证明，则 B_1 中断。否则，B_1 接受 A_1 的证明，则 B_1 发送 α 给可信方并允许可信方回答第二方，注意此时

第二方获得的输出是 $f(\alpha)$，最后 B_1 发给 A_1 期望收到的视图并将 A_1 的输出作为自己的输出。令 p 表示 $A_1(\alpha)$ 说服验证者 $(u,v)\in L$ 的概率，注意这里 $v=A_1(\alpha)$ 不一定等于 $f(\alpha)$，根据 p 与 μ 的关系，分以下两种情况讨论 $\mathrm{Real}_{\Pi,\overline{A}}(\alpha,h(\alpha))$ 和 $\mathrm{Ideal}_{f,\overline{B}}(\alpha,h(\alpha))$ 的关系。

(1) $p>\mu(n)$。在此情形下，根据可靠性条件，必有 $(u,v)\in L$，则有 $A_1(\alpha)=v=f(\alpha)$。因此，以 p 的概率，实际协议和理想模型中的执行不中断，且都等于 $(A_1(\alpha,T),f(\alpha))$，其中 T 表示证明者在第(2)步的视图；以 $1-p$ 的概率，实际协议和理想模型中的执行中断，且都等于 $(A_1(\alpha,T),\perp)$，因此 $\mathrm{Real}_{\Pi,\overline{A}}(\alpha,h(\alpha))$ 和 $\mathrm{Ideal}_{f,\overline{B}}(\alpha,h(\alpha))$ 的分布相同。

(2) $p\leqslant\mu(n)$。在此情形下，实际协议和理想模型中的中断执行以 $1-p$ 的概率发生，且分布相同；由于 $\mu(n)$ 是可忽略的，因此对于非中断执行，实际协议与理想模型之间的差别是可忽略的，这样 $\mathrm{Real}_{\Pi,\overline{A}}(\alpha,h(\alpha))$ 和 $\mathrm{Ideal}_{f,\overline{B}}(\alpha,h(\alpha))$ 是计算不可区分的。

编译器中应用的第三个功能函数是像传输(也称为非认证计算)：$(\alpha,1^{|\alpha|})\mapsto(\lambda,f(\alpha))$，其中函数 f 是多项式时间可计算的。与认证计算不同的是，此处 $f(\alpha)$ 不是可验证的，只要求第二方的输出是函数 f 的像。实现像传输功能函数的协议与协议 6.6 相同，但是要求 (P,V) 是强知识的零知识证明系统，即以下命题 6.5 成立。

命题 6.5 假设 (P,V) 是关系 $R=\{(v,w):v=f(w)\}$ 的强知识的零知识证明系统，h 是一个常函数(不失一般性，设 $h(\alpha)=1^{|\alpha|}$)，则协议 6.6 安全计算像传输功能函数。

证明 与命题 6.4 的证明类似，分两种情况讨论。第一方诚实的情形其证明与命题 6.4 完全相同。因为第二方的视图在协议 6.5 与协议 6.6 中是相同的，所以采用与命题 6.4 中相同的方法，将实际协议中第二方算法 A_2 转换成理想模型中相应的算法 B_2。

第二方诚实。在这种情形下，B_2 是确定的，需要将攻击实际协议的敌手 A_1 转化为攻击理想模型的敌手 B_1，B_1 将 A_1 作为子程序调用，算法 B_1 的具体设计如下：B_1 获得输入 $\alpha\in\{0,1\}^n$，B_1 以输入 α 调用 A_1，记 A_1 发出的消息为 v，即 $v\leftarrow A_1(\alpha)$，B_1 试图获得 v 在 f 下的原像，确切地讲，B_1 应用此证明系统的知识抽取器抽取出串 w，使得 $f(w)=v$，若抽取器成功，B_1 令 $\alpha'=w$，否则，B_1 令 $\alpha'=\perp$。接下来 B_1 要生成 A_1 在第(2)步的视图，B_1 充当诚实的验证者与 A_1 交互，若验证者拒绝 A_1 的证明，则 B_1 中断，并将 A_1 的输出作为自己的输出；若被模仿的验证者接受 A_1 的证明，分成以下两种情况。

(1) 若 $\alpha'\neq\perp$，则 B_1 发送 α' 给可信方并允许可信方回答第二方，注意此时给第二方的回答是 $f(\alpha')$。

(2) 若 $\alpha'=\perp$，则 B_1 中断。最后 B_1 将执行视图发送给 A_1 并将 A_1 的输出作为自己的输出。下面证明 $\mathrm{Real}_{\Pi,\overline{A}}(\alpha,1^{|\alpha|})$ 和 $\mathrm{Ideal}_{f,\overline{B}}(\alpha,1^{|\alpha|})$ 是计算不可区分的，实际上这两个随机变量是统计不可区分的，二者的差别仅在于，攻击实际协议的敌手 A_1 成功地使验证者确信 A_1 知道 v 在 f 下的原像，然而知识抽取器没有能够抽取出这个原像。在其他两种情况下，即 A_1 没能说服验证者他知道原像以及 A_1 说服验证者并且知识抽取器成功，$\mathrm{Real}_{\Pi,\overline{A}}(\alpha,1^{|\alpha|})$ 与 $\mathrm{Ideal}_{f,\overline{B}}(\alpha,1^{|\alpha|})$ 是等分布的。而由知识的零知识证明定义，A_1 说服验证者并且知识抽取器失败的概率是可忽略的，因此 $\mathrm{Real}_{\Pi,\overline{A}}(\alpha,1^{|\alpha|})$ 和 $\mathrm{Ideal}_{f,\overline{B}}(\alpha,1^{|\alpha|})$ 是计算不可区分的。

编译器中应用的第四个功能函数是认证计算，是部分认证计算功能函数的推广。令 $f:\{0,1\}^*\times\{0,1\}^*\rightarrow\{0,1\}^*$ 和 $h:\{0,1\}^*\rightarrow\{0,1\}^*$ 是多项式时间可计算的，且 h 是单射，则 h 认证 f 计算功能函数定义为：$(\alpha,\beta)\mapsto\begin{cases}(\lambda,f(\alpha)), & \text{若 } \beta=h(\alpha)\\ (\lambda,(h(\alpha),f(\alpha))), & \text{其他}\end{cases}$。以下设计的利用像传

输功能函数的协议 6.7 安全计算此功能函数。

协议 6.7（认证计算协议）

输入：第一方输入是 $\alpha\in\{0,1\}^*$，第二方输入是 $\beta\in\{0,1\}^{|\alpha|}$。

(1) 第一方应用像传输功能函数发送 $(u,v)=(h(\alpha),f(\alpha))$ 给第二方。

(2) 第二方收到 (u,v) 后，若 $u=\beta$，则输出 v；否则，输出 (u,v)。

输出：第一方没有输出，第二方若没有中断，则输出如第(2)步。

命题 6.6 协议 6.7 将 h 认证 f 计算功能函数安全归约到像传输功能函数。

证明 与前面的证明类似，分两种情况讨论。第一方诚实，需要将攻击实际协议的敌手 A_2 变换到攻击理想模型的敌手 B_2，算法 B_2 的具体设计如下：B_2 发送 β 给可信方并获得计算结果，若 $\beta=h(\alpha)$，则结果是 $v=f(\alpha)$，若 $\beta\neq h(\alpha)$，则结果是 $(u,v)=(h(\alpha),f(\alpha))$。$\beta=h(\alpha)$ 时，B_2 置 $u=\beta$，这样对于从可信方获得的这两种结果都有 $(u,v)=(h(\alpha),f(\alpha))$。$B_2$ 将 β 和 (u,v) 发给 A_2，并以 A_2 的输出作为输出。易见，$\mathrm{Real}_{\Pi,\bar{A}}(\alpha,\beta)$ 和 $\mathrm{Ideal}_{f,\bar{B}}(\alpha,\beta)$ 是等分布的，都是 $(\lambda,A_2(\beta,(h(\alpha),f(\alpha))))$。

第二方诚实。在这种情形下，B_2 是确定的，需要将攻击实际协议的敌手 A_1 转化为攻击理想模型的敌手 B_1，算法 B_1 的具体设计如下：B_1 以输入 α 调用 A_1，获得 A_1 在第(1)步发出的消息 α'，若 A_1 指令预言器不回答第二方，则 B_1 中断，否则，B_1 发送 α' 给可信方并允许可信方回答第二方。B_1 将执行视图 α',λ 给 A_1 并输出 A_1 的输出，易见，$\mathrm{Real}_{\Pi,\bar{A}}(\alpha,\beta)$ 和 $\mathrm{Ideal}_{f,\bar{B}}(\alpha,\beta)$ 是分布相等的随机变量。

编译器中应用的第五个功能函数是扩展掷币，是掷币入井功能函数的推广。对于任意正多项式 l 和多项式时间可计算函数 g，扩展掷币功能函数定义为：$(1^n,1^n)\mapsto(r,g(r))$，其中 r 在 $\{0,1\}^{l(n)}$ 上均匀分布。利用掷币入井和认证计算功能函数可设计以下的协议 6.8 安全计算此功能函数。

协议 6.8（扩展掷币协议） 对于 $r_1,\cdots,r_l\in\{0,1\}^n$，$\sigma_1,\cdots,\sigma_l\in\{0,1\}$，令 $\bar{C}_{r_1,\cdots,r_l}(\sigma_1,\cdots,\sigma_l)=(C_{r_1}(\sigma_1),\cdots,C_{r_l}(\sigma_l))$。

输入：两方获得的输入都是安全参数 1^n，并设 $l=l(n)$。

(1) 第一方均匀选取 $\sigma_1,\cdots,\sigma_l\in\{0,1\}$ 和 $s_1,\cdots,s_l\in\{0,1\}^n$，并且令 $r'=\sigma_1\cdots\sigma_l$，$\bar{s}=s_1\cdots s_l$。

(2) 第一方应用认证计算功能函数发送 $\bar{c}=\bar{C}_{\bar{s}}(r')$ 给第二方。

(3) 两个参与方调用掷币入井功能函数 l 次，生成一个公共的随机串 $r''\in\{0,1\}^l$，即第 i 次调用产生 r'' 的第 i 个比特。

(4) 第一方令 $r=r'\oplus r''$，并利用认证计算功能函数发送 $g(r)$ 给第二方。确切地讲，第一方以输入 $(r',\bar{s},r'')$，第二方以输入 $(\bar{c},r'')$ 调用认证计算功能函数，其中第二方的输入与第一方输入关系应满足 $h(r',\bar{s},r'')=(\bar{C}_s(r'),r'')$，第二方期望获得 $f(r',\bar{s},r'')=g(r'\oplus r'')$。若第一方中断或者第二方从认证计算功能函数获得形如 $((\bar{C}_s(r'),r''),g(r'\oplus r''))$ 的结果，则第二方中断并输出 $\perp$。

输出：第一方输出 r，第二方输出在第(4)步确定，或者是 $g(r)$ 或者是 $\perp$。

命题 6.7 协议 6.8 将扩展掷币功能函数归约到认证计算功能函数和掷币功能函数。

证明 首先假设第一方是诚实的，需要对于任意攻击实际协议的敌手 A_2，设计攻击理想模型的敌手 B_2。算法 B_2 的设计如下：给定输入 1^n，B_2 均匀选择 $r'\in\{0,1\}^l$ 和 $\bar{s}\in\{0,1\}^{l\cdot n}$，仿

效实际协议中第(2)步的执行,将$\bar{c}=\bar{C}_{\bar{s}}(r')$发给$A_2$(实际协议第(2)步中$A_2$应收到的消息),$B_2$均匀选择$r''\in\{0,1\}^l$并发送$r''$给$A_2$模仿第(3)步$A_2$应收到的消息。$B_2$以输入$1^n$调用可信方得到输出$g(r)$,这里$r$是可信方均匀掷币的结果,并发送给诚实的第一方作为输出。A_2收到第(3)步的消息之后,发送出调用认证计算功能函数的询问,若此询问与B_2之前发给A_2的消息$(\bar{C}_{\bar{s}}(r'),r'')$不一致,则$B_2$中断并输出$A_2(\lambda,\bar{c},r'',((\bar{c},r''),g(r)))$;否则,$B_2$输出$A_2(\lambda,\bar{c},r'',g(r))$。根据$A_2$发送出调用认证计算功能函数的询问,区分两种情况讨论。

第一种情况,若A_2发出的询问与接收到的$(\bar{C}_{\bar{s}}(r'),r'')$一致,则$\text{Real}_{\Pi,\bar{A}}(1^n,1^n)=(r'\oplus r'',A_2(\lambda,\bar{C}(r'),r'',g(r'\oplus r'')))$,其中$r'$与$r''$均匀独立分布,记$r=r'\oplus r''$,将$\text{Real}_{\Pi,\bar{A}}(1^n,1^n)$改写为$(r,A_2(\lambda,\bar{C}(r\oplus r''),r'',g(r)))$,而$\text{Ideal}_{f,\bar{B}}(1^n,1^n)=(r,A_2(\lambda,\bar{C}(r'),r'',g(r)))$,其中$r$、$r'$、$r''$均匀独立分布,根据承诺方案$C$的隐蔽性质可知,这两个随机变量是计算不可区分的。第二种情况,若A_2发出的询问与接收到的$(\bar{C}_{\bar{s}}(r'),r'')$不一致,则$\text{Real}_{\Pi,\bar{A}}(1^n,1^n)=(r,A_2(\lambda,\bar{C}(r\oplus r''),r'',((\bar{C}(r\oplus r''),r'',g(r))))$,而$\text{Ideal}_{f,\bar{B}}(1^n,1^n)=(r,A_2(\lambda,\bar{C}(r'),r'',((\bar{C}(r'),r''),g(r))))$,同样根据承诺方案$C$的隐蔽性质可知,这两个随机变量计算不可区分。

第二方诚实。在这种情形下,B_2是确定的,需要将攻击实际协议的敌手A_1转化为攻击理想模型的敌手B_1,算法B_1的具体设计如下:B_1以输入1^n调用A_1,A_1发出消息$(r',\bar{s})\leftarrow A_1(1^n)$是$A_1$在第(2)步调用认证计算功能函数的询问。$B_1$以输入$1^n$调用可信方获得计算结果,一个均匀分布的比特串$r\in\{0,1\}^l$,模仿协议的第(3)步,$B_1$将$A_1$这一步要收到的消息$r''$,设置为$r''=r\oplus r'$,并发送给$A_1$。$B_1$模仿协议第(4)步的执行,令$(q',\bar{q},q'')=A_1(1^n,\lambda,r'')$,若$(\bar{C}_{\bar{q}}(q'),q'')=(\bar{C}_{\bar{s}}(r'),r'')$,即$A_1$在第(4)步以匹配的消息调用认证计算功能函数,则$B_1$指令可信方将计算结果$g(r)$发送给第二方;否则,不发送。$B_1$输出$A_1(1^n,\lambda,r'',\lambda)$。第一方在理想模型和实际协议中的输出都是$A_1(1^n,\lambda,r'',\lambda)$,关键在于与第二方输出的关系,第二方获得输出在两个模型中有输出的条件是一样的,即当且仅当在第(4)步中,有$(\bar{C}_{\bar{q}}(q'),q'')=(\bar{C}_{\bar{s}}(r'),r'')$成立,而根据承诺方案的约束性质,可得出$(q',q'')=(r',r'')$,这样若上述等式成立,则第二方有输出,且在理想模型和实际协议中的输出都是$g(r'\oplus r'')$,因此$\text{Real}_{\Pi,\bar{A}}(1^n,1^n)$和$\text{Ideal}_{f,\bar{B}}(1^n,1^n)$计算不可区分。

在编译器中直接应用的是扩展掷币的一种特殊形式,即$g(r,s)=\bar{C}_s(r)$,具体地讲,是以下功能函数:$(1^n,1^n)\mapsto((r,s),\bar{C}_s(r))$。

编译器中应用的第六个功能函数是输入承诺功能函数。令C是一个承诺方案,$\bar{C}$如协议6.8中定义,输入承诺功能函数定义为:$(x,1^{|x|})\mapsto(r,\bar{C}_r(x))$,其中$r$在$\{0,1\}^{|x|^2}$上均匀分布。利用掷币入井和认证计算功能函数可设计以下的协议6.9安全计算此功能函数。

协议 6.9 第一方输入是$x\in\{0,1\}^n$,第二方输入是1^n。

(1) 第一方均匀选取$r'\in\{0,1\}^{n^2}$。

(2) 第一方利用认证计算功能函数发送$c'=\bar{C}_{r'}(x)$给第二方。

(3) 生成最终承诺所需要的随机串。双方应用扩展掷币功能函数,第一方获得(r,r''),第二方获得$c''=\bar{C}_{r''}(r)$,其中$r\in\{0,1\}^{n^2}$和$r''\in\{0,1\}^{n^3}$是均匀独立分布的随机串。

(4) 第一方利用认证计算功能函数发送$\bar{C}_r(x)$给第二方。具体地讲,第一方以(x,r,r',r''),第二方以(c',c'')作为给认证计算功能函数的询问,两方询问之间的关系是$h(x,r,r',r'')=$

$(\bar{C}_{r'}(x),\bar{C}_{r''}(r))$，第二方期望获得 $f(x,r,r',r'')=\bar{C}_r(x)$。若第一方中断或第二方由于双方的询问不匹配获得 $((\bar{C}_{r'}(x),\bar{C}_{r''}(r)),\bar{C}_r(x))$，则第二方检测出第一方欺骗，输出⊥，中断协议。

输出：第一方输出 r，第二方输出由第(4)步决定，即或者输出 $\bar{C}_r(x)$ 或者输出⊥。

命题 6.8 协议 6.9 将输入承诺功能函数归约到认证计算功能函数和掷币功能函数。

证明 首先假设第一方是诚实的，需要对于任意攻击实际协议的敌手 A_2，设计攻击理想模型的敌手 B_2。算法 B_2 的设计如下：给定输入 1^n，B_2 均匀选择 $r'\in\{0,1\}^{n^2}$，模仿协议的第(2)步，B_2 将 $c'=\bar{C}_{r'}(0^n)$ 发送给 A_2。模仿协议的第(3)步，B_2 均匀选择 $s\in\{0,1\}^{n^2}$ 和 $r''\in\{0,1\}^{n^3}$，将 $c''=\bar{C}_{r''}(s)$ 发送给 A_2。B_2 以输入 1^n 调用可信方得到 $\bar{C}_r(x)$，此处 r 是均匀分布在 $\{0,1\}^{n^2}$ 上的一个随机串，且可信方将 r 发送给第一方作为输入承诺函数的计算结果。A_2 收到 c'' 之后，产生第(4)步的询问，若 A_2 输出的询问不等于 (c',c'')，则 B_2 中断，并输出 $A_2(\lambda,c',c'',((c',c''),\bar{C}_r(x)))$；否则，$B_2$ 输出 $A_2(\lambda,c',c'',\bar{C}_r(x))$。分两种情况讨论。

第一种情况，第(4)步 A_2 发出的询问与之前收到的消息匹配，此时理想模型中的执行结果是 $(r,A_2(\lambda,\bar{C}(0^n),\bar{C}(s),\bar{C}_r(x)))$，其中 r 和 s 独立均匀分布；而实际协议中的执行结果是 $(r,A_2(\lambda,\bar{C}(x),\bar{C}(r),\bar{C}_r(x)))$，根据承诺方案的隐蔽性质可知，这两个随机变量计算不可区分。

第二种情况，第(4)步 A_2 发出的询问与之前收到的消息不匹配，此时理想模型中的执行结果是 $(r,A_2(\lambda,\bar{C}(0^n),\bar{C}(s),((\bar{C}(0^n),\bar{C}(s)),\bar{C}_r(x)))$，其中 r 和 s 独立均匀分布；而实际协议中的执行结果是 $(r,A_2(\lambda,\bar{C}(x),\bar{C}(r),((\bar{C}(x),\bar{C}(r)),\bar{C}_r(x)))$，根据承诺方案的隐蔽性质可知，这两个随机变量计算不可区分。

第二方诚实。在这种情形下，B_2 是确定的，需要将攻击实际协议的敌手 A_1 转化为攻击理想模型的敌手 B_1，算法 B_1 的具体设计如下：B_1 以输入 x 调用 A_1，A_1 发出消息 $(x',r')\leftarrow A_1(x)$ 是 A_1 在第(2)步调用认证计算功能函数的询问。B_1 以输入 x' 调用可信方获得计算结果，一个均匀分布的比特串 $r\in\{0,1\}^{n^2}$，模仿协议的第(3)步，B_1 均匀选取 $r''\in\{0,1\}^{n^3}$，并发送 (r,r'') 给 A_1。B_1 模仿协议第(4)步的执行，令 $(q_1,q_2,s_1,s_2)=A_1(x,\lambda,(r,r''))$，若 $(\bar{C}_{s_1}(q_1),\bar{C}_{s_2}(q_2))=(\bar{C}_{r'}(x'),\bar{C}_{r''}(r))$，即 A_1 在第(4)步以匹配的消息调用认证计算功能函数，则 B_1 指令可信方将计算结果发送给第二方；否则，不发送。B_1 输出 $A_1(x,\lambda,(r,r''),\lambda)$。第一方在理想模型和实际协议中的输出都是 $A_1(x,\lambda,(r,r''),\lambda)$，关键在于与第二方输出的关系，在两个模型中第二方获得输出的条件是一样的，即当且仅当在第(4)步中，有 $(\bar{C}_{s_1}(q_1),\bar{C}_{s_2}(q_2))=(\bar{C}_{r'}(x'),\bar{C}_{r''}(r))$ 成立，而根据承诺方案的约束性质，可得出 $(q_1,q_2)=(x',r)$，这样若上述等式成立，则第二方有输出，且在理想模型和实际协议中的输出都是 $\bar{C}_r(x')$，因此 $\mathrm{Real}_{\Pi,\bar{A}}(x,1^n)$ 和 $\mathrm{Ideal}_{f,\bar{B}}(x,1^n)$ 计算不可区分。

6.3.3 编译器

编译器的工作方式是，给定一个半诚实模型中安全的协议 Π，生成恶意模型中的协议 Π'，Π' 和 Π 完成同样的功能。以下的协议 6.10 即为经过编译的协议的具体结构。

协议 6.10 给定半诚实模型的协议 Π，以下产生恶意模型中的协议 Π'：

输入：第一方的输入是 $x\in\{0,1\}^n$，第二方的输入是 $y\in\{0,1\}^n$。

输入承诺阶段:参与协议的双方利用输入承诺功能函数分别对自己的输入作承诺。具体地,双方两次调用输入承诺功能函数。第一次调用是第一方对自己的输入作承诺,预言器返回给第一方的回答是一个均匀分布的随机串 $\rho^1 \in \{0,1\}^{n^2}$,第二方得到的回答是 $\gamma^1 = \overline{C}_{\rho^1}(x)$。第二次调用是第二方对自己的输入作承诺,预言器返回给第二方的回答是一个均匀分布的随机串 $\rho^2 \in \{0,1\}^{n^2}$,第一方得到的回答是 $\gamma^2 = \overline{C}_{\rho^2}(y)$。

随机带生成阶段:参与协议的双方利用扩展掷币功能函数分别产生自己的随机带。具体地,双方两次调用扩展掷币功能函数。第一次调用是第一方用于获得自己的随机带,预言器返回给第一方的回答是 $(r^1, \omega^1) \in \{0,1\}^{l(n)} \times \{0,1\}^{n \cdot l(n)}$,第二方得到的回答是 $\delta^1 = \overline{C}_{\omega^1}(r^1)$。第二次调用是第二方获得自己的随机带,预言器返回给第二方的回答是 $(r^2, \omega^2) \in \{0,1\}^{l(n)} \times \{0,1\}^{n \cdot l(n)}$,第一方得到的回答是 $\delta^2 = \overline{C}_{\omega^2}(r^2)$。

协议执行阶段:参与协议的双方利用认证计算功能函数模仿协议 Π 每一步的执行。在每次调用中,发送方充当调用预言器的第一方,接收方充当调用预言器的第二方。假设当前步骤是第 j 方发送消息,若 $j=1$,令 $u=x$;若 $j=2$,$u=y$。调用认证计算功能函数的输入 α、β 以及函数 h、f 的定义如下。$\alpha = (\alpha_1, \alpha_2, \alpha_3)$,其中 $\alpha_1 = (u, \rho^j)$ 是调用输入承诺功能函数的询问及回答,$\alpha_2 = (r^j, \omega^j)$ 是调用扩展掷币功能函数的回答,α_3 是第 j 方执行到当前步骤所获得的消息序列。$\beta = (\gamma^j, \delta^j, \alpha_3)$,其中 γ^j 是调用输入承诺功能函数的回答,δ^j 是调用扩展掷币功能函数的回答,α_3 是接收方执行到当前步骤所发出的消息序列。函数 h 定义为 $h(\alpha_1, \alpha_2, \alpha_3) = (\overline{C}_{\rho^j}(u), \overline{C}_{\omega^j}(r^j), \alpha_3)$。函数 f 定义为 $f(\alpha_1, \alpha_2, \alpha_3)$ 即为当前步骤第 j 方要发送的消息。

输出:一旦某个功能函数调用以中断结束,则接收到 $\perp$ 的参与方输出 $\perp$,否则,输出为此参与方在协议 Π 中的输出。

如果将协议 6.10 中对功能函数的调用替换成相应的子协议,那么就得出抵抗恶意敌手的标准协议。显然,若两个参与方都是诚实的,则协议 Π' 完全保持了协议 Π 的输入/输出之间的关系。对于恶意敌手,这个结果也成立,即对于任意功能函数 f,编译器将半诚实模型中保密计算 f 的协议转化为恶意模型中安全计算 f 的协议。换句话说,只要证明协议 6.10 安全计算协议 Π 的功能函数,则定理 6.3 得证。限于篇幅,此处省略,具体证明过程可参见文献[1]。至此,建立了两方计算的一般结论,即假设加强陷门单向置换族存在,则任意功能函数都可以安全计算。6.4 节将两方情况推广到多方情况。

6.4 安全多方计算

本节讨论安全多方计算,其最终目标仍然是设计抵抗任意可行敌手攻击的协议,方法类似于两方的情形,首先对半诚实敌手设计协议,然后对恶意敌手设计协议。对于恶意敌手,多方计算较之两方情形复杂,要考虑两种不同的模型。第一种恶意行为模型类似于两方情形,在这种模型中,敌手可以入侵多数参与方,在安全性定义中允许中断执行。第二种恶意行为模型中,敌手只能控制严格少数的参与方,在安全性定义中可有效防止中断执行。

6.4.1 安全多方计算的定义

一个多方协议可以看作将输入序列映射到输出序列的随机过程。设 m 表示参与方的个

数，为简便起见，不妨设 m 是固定的。一个 m 元功能函数记为 $f:(\{0,1\}^*)^m \to (\{0,1\}^*)^m$，是将序列 $\bar{x}=(x_1,\cdots,x_m)$ 映射到随机变量序列 $f(\bar{x})=(f_1(\bar{x}),\cdots,f_m(\bar{x}))$ 的随机过程，第 i 方的输入是 x_i，期望获得 $f(x_1,\cdots,x_m)$ 的第 i 个位置的分量 $f_i(x_1,\cdots,x_m)$。

对于两方计算的情形，将参与方之一作为敌手，而对于多方计算，引进外部敌手的概念，即存在外部敌手，控制不诚实参与方的集合。敌手可以控制任意个数的参与方。非自适应敌手在协议执行之前确定要入侵的参与方集合，而自适应敌手在协议执行过程中，利用收集到的信息选择要入侵的参与方集合，本节讨论的敌手是非自适应情况。关于通信信道的假设是，外部敌手可以搭线窃听所有的通信信道，特别是诚实方之间的通信。

半诚实模型的定义类似于两方计算的情形。半诚实参与方是指正确执行协议，但是会记录中间运算的结果。

定义 6.8（多方保密计算，无搭线窃听）　设 $f:(\{0,1\}^*)^m \to (\{0,1\}^*)^m$ 是一个 m 元功能函数，$f_i(x_1,\cdots,x_m)$ 表示 $f(x_1,\cdots,x_m)$ 的第 i 个分量。对于 $I=\{i_1,\cdots,i_t\}\subseteq[m]=\{1,\cdots,m\}$，令 $f_I(x_1,\cdots,x_m)$ 表示子序列 $f_{i_1}(x_1,\cdots,x_m),\cdots,f_{i_t}(x_1,\cdots,x_m)$。$\Pi$ 是计算 f 的 m 方协议。第 i 方执行协议过程中的视图如定义 6.1，记为 $\mathrm{View}_i^\Pi(\bar{x})$。对于 $I=\{i_1,\cdots,i_t\}$，令 $\mathrm{View}_I^\Pi(\bar{x})=(I,\mathrm{View}_{i_1}^\Pi(\bar{x}),\cdots,\mathrm{View}_{i_t}^\Pi(\bar{x}))$。当功能函数 f 是确定性函数时，称协议 Π 保密计算 f，如果存在概率多项式时间算法 S，使得对任意 $I\subseteq[m]$，有

$$\{S(I,(x_{i_1},\cdots,x_{i_t}),f_I(\bar{x}))\}_{\bar{x}\in(\{0,1\}^*)^m} \stackrel{c}{\equiv} \{\mathrm{View}_I^\Pi(\bar{x})\}_{\bar{x}\in(\{0,1\}^*)^m}$$

一般地，称协议 Π 为保密计算 f，如果存在概率多项式时间算法 S，使得对任意 $I\subseteq[m]$，有

$$\{(S(I,(x_{i_1},\cdots,x_{i_t}),f_I(\bar{x})),f(\bar{x})\}_{\bar{x}\in(\{0,1\}^*)^m}$$
$$\stackrel{c}{\equiv}\{(\mathrm{View}_I^\Pi(\bar{x}),\mathrm{Output}^\Pi(\bar{x}))\}_{\bar{x}\in(\{0,1\}^*)^m}$$

其中 $\mathrm{Output}^\Pi(\bar{x})$ 表示所有参与方的输出序列。

若对于敌手可以搭线窃听的情况，只需在 $\mathrm{View}_I^\Pi(\bar{x})$ 中包含诚实方之间的通信即可。

下面讨论抵抗恶意敌手攻击的安全性定义。根据敌手控制参与方的个数分为两种情况，第一种情况是对敌手控制参与方的个数不加限制，对于这种情况，安全计算任意功能函数的协议设计完全类似于两方情形，并且安全性定义允许协议中断。第二种情况是敌手控制参与方个数严格小于一半，对于这种情况，安全计算任意功能函数的协议设计比两方情形简单，并且安全性要求协议不能中断。

第一类恶意模型。类似于两方情形。在第一类恶意模型中，有以下 3 种攻击行为不可避免。

(1) 敌手控制的恶意参与方拒绝参与协议运行，与两方情形相同，在多方计算协议中，将这种行为看做一种特殊的输入替换。

(2) 恶意参与方替换输入，执行协议所用的输入与外界提供的输入不同。

(3) 恶意参与方中断协议执行。

相应地，在理想模型中，尽管引进了可信方这 3 种行为也是不能避免的，同样赋予恶意的第一方叫停可信方的权利，即当敌手控制第一方时，能够阻止可信方发送计算结果给其他参与方。

定义 6.9（第一类恶意模型所对应的理想模型）　设 $f:(\{0,1\}^*)^m \to (\{0,1\}^*)^m$ 是一个 m 元功能函数。对于 $I=\{i_1,\cdots,i_t\}\subseteq[m]=\{1,\cdots,m\}$，令 $\bar{I}=[m]\backslash I$，$(x_1,\cdots,x_m)_I=$

$(x_{i_1},\cdots,x_{i_t})$。用(I,B)表示理想模型中的敌手，其中 $I\subseteq[m]$，B 是概率多项式时间算法。理想模型中 f 关于(I,B)的联合执行记为 $\mathrm{Ideal}^{(1)}_{f,I,B(z)}(\bar{x})$，如下定义：

(1) 第一方是诚实方，则 $\mathrm{Ideal}^{(1)}_{f,I,B(z)}(\bar{x})$为$(f_{\bar{I}}(\bar{x}'),B(\bar{x}_I,I,z,r,f_I(\bar{x}')))$，其中$\bar{x}'=(x_1',\cdots,x_m')$，使得对于 $i\in I$，有 $x_i'=B(\bar{x}_I,I,z,r)_i$，对于 $i\notin I$，有 $x_i'=x_i$。

(2) 第一方不诚实，则 $\mathrm{Ideal}^{(1)}_{f,I,B(z)}(\bar{x})$为

$$\begin{cases}(\perp^{|\bar{I}|},B(\bar{x}_I,I,z,r,f_I(\bar{x}'),\perp)) & \text{若 } B(\bar{x}_I,I,z,r,f_I(\bar{x}'))=\perp \\ (f_{\bar{I}}(\bar{x}'),B(\bar{x},I,z,r,f_I(\bar{x}'))) & \text{若 } B(\bar{x}_I,I,z,r,f_I(\bar{x}'))\neq\perp\end{cases}$$

其中$\bar{x}'=(x_1',\cdots,x_m')$使得对于 $i\in I$，有 $x_i'=B(\bar{x}_I,I,z,r)_i$，对于 $i\notin I$，有 $x_i'=x_i$。

定义 6.10（第一类恶意模型中的安全性） 设功能函数 f 如定义 6.9 中给出的一样，协议 Π 是计算 f 的 m 方协议。实际协议中 Π 关于(I,A)的联合执行记为 $\mathrm{Real}_{\Pi,I,A(z)}(\bar{x})$，定义成 m 个参与方交互产生的输出对，其中恶意参与方产生的消息根据 $A(\bar{x}_I,I,z)$计算，诚实方产生的消息根据协议 Π 的指令计算。称协议 Π 在第一类恶意模型中安全计算功能函数 f，如果对于实际协议中任意概率多项式时间算法 A，都存在理想模型中概率多项式时间算法 B，使得对任意 $I\subseteq[m]$，有$\{\mathrm{Ideal}^{(1)}_{f,I,B(z)}(\bar{x})\}_{\bar{x},z}\overset{c}{\equiv}\{\mathrm{Real}_{\Pi,I,A(z)}(\bar{x})\}_{\bar{x},z}$。

注：此处理想模型敌手 B 控制的参与方集合与攻击实际协议敌手 A 控制的参与方集合相同。

定义 6.11（第二类恶意模型中的安全性） 设功能函数 f 如定义 6.9 中给出的一样，协议 Π 是计算 f 的 m 方协议。理想模型敌手除了不允许中断之外，完全与定义 6.9 中一样，理想模型中 f 关于(I,B)的联合执行记为 $\mathrm{Ideal}^{(2)}_{f,I,B(z)}(\bar{x})$，实际协议中 Π 关于(I,A)的联合执行记为 $\mathrm{Real}_{\Pi,I,A(z)}(\bar{x})$，定义为 m 个参与方交互产生的输出对，其中恶意参与方产生的消息根据 $A(\bar{x}_I,I,z)$计算，诚实方产生的消息根据协议Π 的指令计算。称协议 Π 在第二类恶意模型中安全计算功能函数 f，如果对于实际协议中任意概率多项式时间算法 A，都存在理想模型中概率多项式时间算法 B，使得对任意 $I\subseteq[m]$，$|I|<m/2$，有$\{\mathrm{Ideal}^{(2)}_{f,I,B(z)}(\bar{x})\}_{(\bar{x},z)}\overset{c}{\equiv}\{\mathrm{Real}_{\Pi,I,A(z)}(\bar{x})\}_{\bar{x},z}$。

6.4.2　半诚实模型中的安全性

半诚实模型中的安全多方计算协议是安全两方计算协议的直接推广。考虑计算 m 元功能函数的 GF(2)上的算术电路，每个参与方将自己的输入与其他参与方分享，使得所有这些分享的和等于输入比特。对于电路的每一个门，应用适当的保密计算协议，将输入的分享转化为输出的分享，使得输出的这些分享值的和即为经过该电路门的计算结果。类似于两方计算，关键问题是乘法门的计算。确切地讲，电路中的某个乘法门要计算的是 $a\cdot b=c$，那么首先将输入 a 和 b 在 m 个参与方之间分享，使得第 i 方持有的关于这个乘法门的输入的分享值是(a_i,b_i)，满足 $\sum_{i=1}^{m}a_i=a$，$\sum_{i=1}^{m}b_i=b$，然后设计协议，保密计算以下的 m 元功能函数：$((a_1,b_1),\cdots,(a_m,b_m))\mapsto(c_1,\cdots,c_m)$，其中$(c_1,\cdots,c_m)$在$\{0,1\}^m$ 上均匀分布，且服从 $\sum_{i=1}^{m}c_i=\sum_{i=1}^{m}a_i\cdot\sum_{i=1}^{m}b_i$。这样，对于多方保密计算而言，关键问题就是设计协议保密计算乘法，主要思路是将多方乘法保密归约到两方乘法功能函数，这里归约的定义如两方情形的定义。

称 $\sum_{i=1}^{m} c_i = \sum_{i=1}^{m} a_i \cdot \sum_{i=1}^{m} b_i$ 为 m 方乘法功能函数。

协议 6.11（保密归约 m 方乘法功能函数到两方乘法功能函数）

输入：第 i 方的输入是 $(a_i, b_i) \in \{0,1\} \times \{0,1\}, i=1,\cdots,m$。

(1) 每对参与方 (i,j)，其中 $i<j$，调用两方乘法功能函数。第 i 方提供输入对 (a_i, b_i)，第 j 方提供输入对 (a_j, b_j)，将两方乘法功能函数返回给第 i 方的回答记为 $c_i^{\{i,j\}}$，给第 j 方的回答记为 $c_j^{\{i,j\}}$。

(2) 第 i 方置 $c_i = m a_i b_i + \sum_{j \neq i} c_i^{\{i,j\}}$。

输出：第 i 方输出 c_i。

根据等式 $\left(\sum_{i=1}^{m} a_i\right) \cdot \left(\sum_{i=1}^{m} b_i\right) = \sum_{i=1}^{m} a_i b_i + \sum_{1 \leqslant i < j \leqslant m} (a_i b_j + a_j b_i) = m \cdot \sum_{i=1}^{m} a_i b_i + \sum_{1 \leqslant i < j \leqslant m} (a_i + a_j) \cdot (b_i + b_j)$，由协议 6.11 计算 m 方乘法功能函数，下面命题 6.9 表明协议 6.11 是保密归约。

命题 6.9　协议 6.11 保密归约 m 方乘法功能函数到两方乘法功能函数。

证明　给定集合 $I = \{i_1, \cdots, i_t\}$，其中 $t<m$，以及输入序列 $(a_{i_1}, b_{i_1}), \cdots, (a_{i_t}, b_{i_t})$ 和输出序列 $(c_{i_1}, \cdots, c_{i_t})$，模拟器算法 S 构造如下：对每个对 $(i,j) \in I \times I, i<j$，模拟器 S 均匀选择 $c_i^{\{i,j\}} \in \{0,1\}$，并置 $c_j^{\{i,j\}} = c_i^{\{i,j\}} + (a_i + a_j) \cdot (b_i + b_j)$。令 $\bar{I} = [m] \backslash I$，以 l 记 $\bar{I}$ 中的最大元，对于 $i \in I$ 且 $j \in \bar{I} \backslash l$，$S$ 均匀选择 $c_i^{\{i,j\}} \in \{0,1\}$；对于 $i \in I$，S 置 $c_i^{\{i,l\}} = c_i + m a_i b_i + \sum_{j \notin \{i,l\}} c_i^{\{i,j\}}$。对于 $i \in I$ 和 $j \in [m] \backslash \{i\}$，$S$ 输出所有的 $c_i^{\{i,j\}}$。易见，模拟器 S 的输出分布与 I 中参与方的视图分布相等。

多方电路赋值协议。有了计算 m 方乘法功能函数的协议，下面需要设计协议将计算任意确定性功能函数归约到计算 m 方乘法功能函数，思路类似于两方计算的情形，如下协议 6.12 是具体的构造。

协议 6.12（将 m 方确定性功能函数归约到 m 方乘法功能函数）

输入：第 i 方持有输入 $x_i = x_i^1 \cdots x_i^n \in \{0,1\}^n, i=1,\cdots,m$。

(1) 分享输入。每个参与方将自己的输入与其他参与方分享。对任意 $i=1,\cdots,m$ 和 $j=1,\cdots,n$，对任意 $k \neq i$，第 i 方均匀选择比特 $r_k^{(i-1)n+j}$ 并发送给第 k 方作为第 k 方关于第 $(i-1) \cdot n+j$ 条输入线的分享值。第 i 方将自己关于第 $(i-1) \cdot n+j$ 条输入线的分享值设置为 $x_i^j + \sum_{k \neq i} r_k^{(i-1)n+j}$。

(2) 电路赋值。根据电路的线路顺序，参与方利用关于这两条输入线路的各自的分享值，保密计算这个门的输出的分享。参与方分别持有某个门两条输入线的分享，即第 i 方持有分享值 $a_i, b_i, i=1,\cdots,m$，其中 $a_1,\cdots,a_m$ 是第一条输入线的分享，$b_1,\cdots,b_m$ 是第二条输入线的分享。

加法门赋值：第 i 方将加法门的输出线的分享设置为 $a_i + b_i$。

乘法门赋值：参与方以各自关于输入的分享值 (a_i, b_i) 调用 m 方乘法功能函数，以函数返回的回答作为乘法门输出的各自的分享值。

(3) 恢复输出。一旦整个电路的输出线的分享确定，则每个参与方将每条输出线的分享值发送到与之分享的参与方相应的输出线，将每条输出线上获得的计算结果相加，即确定

出每条输出线上的比特。

输出：将输出线上的比特输出。

应用与两方电路赋值协议类似的证明方法，可以证明协议 6.12 实现了由 m 方确定性功能函数到 m 方乘法功能函数的归约。至此，将任意 m 元功能函数归约到两方乘法函数，而后者可归约到健忘传输协议 OT_1^4，因此有以下定理。

定理 6.5 假设存在加强陷门置换族，则任意 m 方功能函数在半诚实模型中都可以保密计算。

6.4.3 恶意模型中的安全性

本小节的目的是建立关于安全多方计算的一般性结论，即以下的定理 6.6。

定理 6.6 假设存在加强陷门置换族，网络中存在公钥基础设施，则任意 m 方功能函数在两类恶意模型中都可以安全计算。

所谓公钥基础设施是初始假设，它为每个参与方生成某个签名方案的密钥对，并且将验证密钥公开。完成定理证明分为两步。首先，将半诚实模型中安全的协议编译为第一类恶意模型中安全的协议，这里的编译器是两方情形的直接推广；其次，将第一类恶意模型中安全的协议编译为第二类恶意模型中安全的协议，此处编译器利用的主要工具是可验证秘密共享协议。为简化表述，本小节假设参与方之间的通信是广播信道，参与方之间没有点到点的通信信道。所谓的广播信道是指在通信的每一个步骤，只有一个参与方发送消息，所有的参与方都收到这个消息，需要强调的是，在确定的步骤，只有协议指定的那个参与方才能够发送消息，即其他的参与方都能够验证当前步骤收到的消息的确是指定的参与方发出的。这样的广播信道可以通过拜占庭协商(Byzantine Agreement)协议在标准的点到点信道上实现。

编译器的目的就是将点到点的半诚实模型中安全的协议转化成两类广播的恶意模型中安全的协议，具体步骤如下。首先应用所谓的前编译器，将点到点模型中保密计算某理想功能函数 f 的协议 Π_0 转化成广播信道模型中保密计算理想功能函数 f 的协议 Π'_0；其次应用第一编译器将 Π'_0 转化为第一类恶意模型中安全的协议 Π'_1，注意这里 Π'_0 和 Π'_1 的通信信道是广播的；再其次，应用第二编译器将 Π'_1 转化为第二类恶意模型中安全的协议 Π'_2，这里 Π'_2 的通信信道是广播的；最后应用后编译器分别将广播信道中安全的协议 Π'_1 和 Π'_2 转化为标准的点到点信道上安全的协议 Π_1 和 Π_2。

下面首先讨论前编译器问题。

命题 6.10 假设存在加强陷门置换族，则任意 m 元功能函数都能够在广播通信模型中保密计算。

证明 令 f 是一个 m 元功能函数，Π 是点到点通信模型中保密计算 f 的协议。给定一个陷门置换，可以构造安全的公钥加密方案，利用此加密方案设计协议 Π' 在广播信道模型中保密计算 f。Π' 中每个参与方生成一对加密、解密密钥，将加密密钥广播。若依协议 Π 指令任意参与方要发送消息 ω 给第 i 方，则利用第 i 方广播的加密密钥加密消息 ω，将密文广播，第 i 方应用解密密钥恢复消息 ω，并按照协议指令继续执行。对于协议 Π'，模拟器算法 S' 设计如下，由于 Π 是点到点通信模型中保密计算 f 的协议，因此存在模拟器算法 S 能够模拟出协议 Π 的视图，S' 利用 S 产生 Π 的视图，之后对于敌手控制的半诚实方发送和收到

的消息，因为 S' 知道相应的消息和加密密钥，S' 如协议 Π' 中一样加密这些消息即可。而对于 S' 不知道的，即 S 不能产生出来的诚实方之间传递的消息，S' 生成对于任意消息比如全零串的加密，这样就完成了对于协议 Π' 视图的模拟。如果能够区分 Π' 的视图与 S' 的输出，那么或者能够区分 Π 的视图与 S 的输出，或者区分真实消息和任意选定消息的加密，这两种情况都不可能，因此命题得证。

接下来讨论后编译器问题。所谓后编译器是指将广播模型中的协议转化成点到点模型中的协议。给定两类恶意模型中的一个安全计算某个功能函数的协议，此协议的通信信道是广播的，通过后编译器将它转化成点到点模型中的安全协议，因此要完成的任务是在有恶意参与方存在的情况下，在点到点的通信信道上实现广播的功能，问题归约到解决认证拜占庭协商问题。

认证拜占庭协商问题是指第一方有输入比特 σ，目标是使所有诚实参与方就 σ 的值达成一致，若第一方诚实，则所有诚实参与方达成一致的值是 σ；而若第一方不诚实，则所有诚实参与方也能够达成一致，不过这时达成一致的值可能不是 σ。这里解决认证拜占庭协商问题的基础是同步的点到点的通信模型和签名基础设施，即每个参与方知道其他参与方的验证密钥。

为叙述方便，引入 (v,i) 认证消息的概念。如果一个消息具有形式 $(v,s_{p_1},\cdots,s_{p_i})$，其中 $p_1=1$，其他所有的 p_j 互不相同，并对每一个 $j=1,\cdots,i$，串 s_{p_j} 是关联于第 p_j 方的验证密钥的消息 $(v,s_{p_1},\cdots,s_{p_{j-1}})$ 的签名，则称此消息是 (v,i) 认证的，也称 (v,i) 认证消息。

协议 6.13（认证拜占庭协商）

设 m 表示参与方个数。假定所使用的签名算法的签名长度只依赖于安全参数，而不依赖于被签消息的长度。

阶段 1：第 1 方签名他的输入，并将得到的输入-签名对发送给所有其他参与方。第 1 方有可能在此结束协议。

阶段 $\boldsymbol{i(i=2,3,\cdots,m)}$：每个诚实的参与方（除了第 1 方）检查他在第 $i-1$ 阶段收到的消息，并转发他所收到的第 $(\cdot,i-1)$ 认证签名版本。具体来讲，对每一个 $v\in\{0,1\}$，如果第 j 方已经收到了一个 $(v,i-1)$ 认证消息 $(v,s_{p_1},\cdots,s_{p_{i-1}})$ 满足所有的 p_k 都不同于 j，则他把签名附到消息上，并发送所得的结果 (v,i) 认证消息给所有的参与方。

这里强调，对每一个 v 的值，第 j 方至多发送一个 (v,i) 认证消息给所有的参与方。实际上，对某个 $i'<i$，如果他已经发送了 (v,i') 认证消息，就不必再发送 (v,i) 认证消息。

输出：每个诚实的参与方（除了第 1 方）按下列方式输出：

(1) 如果对某 $i_0,i_1\in[m]$（未必不同），他既收到了一个 $(0,i_0)$ 认证消息，又收到了一个 $(1,i_1)$ 认证消息，则他断定第 1 方是恶意的，并输出错误标志 $\perp$。

(2) 如果对一个 $v\in\{0,1\}$ 和某个 i，他收到了一个 (v,i) 认证消息，则输出 v。

(3) 如果对任意的 $v\in\{0,1\}$ 和 i，他没有收到一个 (v,i) 认证消息，则他断定第 1 方是恶意的，并输出错误标志 $\perp$。

显然，在协议 6.13 中，当第 1 方以输入 v 开始执行协议时，在阶段 1，他给每一个参与方发送一个 $(v,1)$ 认证消息。对每一个 $i\geqslant 2$，如果 $(v,s_{p_1},\cdots,s_{p_i})$ 是 (v,i) 认证的，则 $(v,s_{p_1},\cdots,s_{p_{i-1}})$ 是 $(v,i-1)$ 认证的。

下面以认证拜占庭协商为基本模块来解决后编译器问题。

命题 6.11(后编译器) 假设单向函数存在。若存在公钥基础设施,则任意恶意广播模型中能够被安全计算的 m 元功能函数,也能够在点到点的恶意模型中安全计算。

证明 首先将广播模型中的协议转化成点到点模型中的协议,转化方式是原协议中发送的每一个广播消息,用认证拜占庭协商协议替换,以实现点到点的等价协议。安全性证明是将攻击点到点协议的任意实际敌手,转化成相应的攻击广播信道中协议的实际敌手,因为广播信道中的协议是安全的,从而点到点信道中的协议是安全的。具体地,除了通信模型改变,敌手其他攻击不变,而广播信道上广播的任何一个消息都以认证拜占庭协商协议的执行所替代,根据签名方案的不可伪造性质,保证所有诚实方收到相同的消息,这与广播信道的功能一致,从而命题得证。

根据前编译器和后编译器的功能,以下讨论协议的安全性时,不再考虑通信模型。为了表示简单,都假设通信信道是广播的,即所有的诚实方收到同样的消息,若此消息是诚实方发送的,则所有诚实方收到同样的值,若此消息是敌手控制的参与方发送的,则所有诚实方收到同样的值或者同样检测出欺骗,中断执行。总之,对于任一个消息,所有的诚实方都能够达成一致意见。

6.4.4 第一类编译器

第一类编译器的构造方式基本类似于两方情形,只不过是将两方情形的编译器调用的两方功能函数替换成 m 元功能函数,有与两方情形完全相同的归约定理,此处不再赘述。注意当前讨论协议的通信信道是广播的,对于搭线窃听的敌手来说,不通过控制参与方只是通过广播信道就能够窃听到消息。为了抵抗这种敌手的攻击,需要以下多方特有的秘密广播功能函数:$(\alpha,1^{|\alpha|},\cdots,1^{|\alpha|})\mapsto(\alpha,\alpha,\cdots,\alpha)$。假设陷门置换存在,则可以构造公钥加密方案,第一方利用其他参与方的公钥分别对 α 加密并且广播出去,为避免恶意的第一方向不同的参与方发送不同的消息,利用认证拜占庭协商协议传递加密后的密文。第一类编译器需要的功能函数分别是多方输入承诺功能函数,多方扩展掷币功能函数以及多方认证计算功能函数,其中多方认证计算功能函数要利用多方像传输功能函数实现。

多方像传输功能函数:$(\alpha,1^{|\alpha|},\cdots,1^{|\alpha|})\to(\lambda,f(\alpha),\cdots,f(\alpha))$。关于多方像传输功能函数的安全实现意味着或者所有参与方得到同样的函数值或者检测出第一方欺骗,这样所有的诚实方关于第一方的行为达成一致。

协议 6.14(多方像传输协议) 令关系 $R=\{(v,w):v=f(w)\}$。

输入:第一方输入是 $\alpha\in\{0,1\}^*$,其他参与方的输入是 $1^{|\alpha|}$。

(1) 第一方秘密广播 $v=f(\alpha)$。

(2) 对于 $i=2,\cdots,m$,第一方和第 i 方调用关系 R 的零知识强知识证明系统 (P,V),第一方作为证明者,第 i 方作为验证者。证明系统的公共输入是 v,证明者的辅助输入是 α,要向验证者证明他知道 w,使得 $(v,w)\in R$。一旦验证者拒绝证明,第 i 方广播他在证明中应用的随机带,使得其他参与方确信第 i 方是合理拒绝证明。

输出:对于 $i=2,\cdots,m$,若第 i 方收到一个合理拒绝,则输出 $\perp$,否则输出 v。

命题 6.12 假设证明系统 (P,V) 是关系 $R=\{(v,w):v=f(w)\}$ 的强知识的零知识证明系统,则协议 6.14 安全计算像传输功能函数。

证明 证明是两方情形的自然推广。将实际协议中敌手控制的参与方集合记为 I,并

作为辅助输入提供给实际协议的敌手和理想模型敌手。$I=\varnothing$时，根据秘密广播的安全性，协议 6.14 也是安全的。$I\neq\varnothing$时，分两种情况讨论。

第一方诚实，即 $1\notin I$。在这种情形下，算法 B 的具体设计如下：B 代表 I 中每个参与方，发送输入 $1^{|\alpha|}$ 给可信方并得到输出 v。对于 $i=2,\cdots,m$，B 调用零知识证明系统的模拟器算法，提供给模拟器的输入是 v，将 A 作为恶意的验证者算法，一个模拟过程结束，将模拟过程交给 A，这部分模拟视图记为 $S=S(v)$。B 将视图(v,S)给 A，并以 A 的输出作为输出。实际协议的敌手的输出与理想模型的敌手的输出是计算不可区分的，这是由零知识证明模拟器的性质直接保证的，而对于诚实方来说，在实际协议和理想模型中的输出都是 $v=f(\alpha)$，因此这两个随机变量是计算不可区分的。

第一方不诚实，即 $1\in I$。在这种情形下，算法 B 的具体设计如下：B 获得输入 $\alpha\in\{0,1\}^n$，B 以输入 α 调用 A，记 A 发出的消息为 v，即 $v\leftarrow A(\alpha)$，B 试图获得 v 在 f 下的原像，确切地讲，B 应用此证明系统的知识抽取器抽取出串 w，使得 $f(w)=v$，此抽取器对于 $|\bar{I}|$ 个证明系统尝试抽取，若抽取器成功，B 令 $\alpha'=w$；否则，B 令 $\alpha'=\perp$。接下来 B 要生成 A 在第(2)步的视图，对于每个 $i\in\bar{I}$，B 充当诚实的验证者与 A 交互，若 $m-1$ 个被模仿的验证者中有一个合理拒绝，则 B 中断，并将 A 的输出作为自己的输出；若被模仿的验证者都接受 A 的证明，分成以下两种情况：

(1) 若 $\alpha'\neq\perp$，则 B 发送 α' 给可信方并允许可信方回答诚实方，注意此时给诚实方的回答是 $f(\alpha')$。

(2) 若 $\alpha'=\perp$，则 B 中断。最后 B 将执行视图发送给 A 并将 A 的输出作为自己的输出。

与两方情形相同，实际协议的执行与理想模型的模拟二者之间的差别仅在于，攻击实际协议的敌手 A 成功地使验证者确信 A 知道 v 在 f 下的原像，然而知识抽取器没有能够抽取出这个原像。而根据知识的零知识证明定义，A 说服验证者并且知识抽取器失败的概率是可忽略的，因此以上两个随机变量是统计不可区分的。

在像传输功能函数的基础上，通过以下协议 6.15 可以实现认证计算功能函数。

协议 6.15(多方认证计算)

输入：第一方输入是 $\alpha\in\{0,1\}^*$，其他参与方输入是 $\beta_i\in\{0,1\}^{|\alpha|}$。

(1) 第一方应用像传输功能函数发送$(u,v)=(h(\alpha),f(\alpha))$给其他参与方。

(2) 其他参与方收到(u,v)后，若 $u=\beta_i$，则输出 v；否则，输出(u,v)。

输出：第一方没有输出，其他参与方若没有中断，则输出如第(2)步。

协议 6.15 的安全性证明是两方情形命题 6.6 的自然推广，此处省略。

第一类编译器中要强制各个参与方使用均匀分布的随机串，类似于两方情形，此处应用多方扩展掷币功能函数达到这一目的，以下的协议 6.16 安全计算多方扩展掷币功能函数，安全性证明沿袭两方扩展掷币功能函数的证明思想，此处不再赘述。

协议 6.16 设 C 是一个承诺方案，且 $\bar{C}_{r_1,\cdots,r_l}(\sigma_1,\cdots,\sigma_l)=(C_{r_1}(\sigma_1),\cdots,C_{r_l}(\sigma_l))$。

输入：每个参与方获得的输入都是安全参数 1^n，并设 $l=l(n)$。

(1) 对于 $i=1,\cdots,m$，第 i 方均匀选取 $r_i\in\{0,1\}^l$ 和 $s_i\in\{0,1\}^{l\cdot n}$。

(2) 对于 $i=1,\cdots,m$，第 i 方应用认证计算功能函数发送 $c_i=\bar{C}_{s_i}(r_i)$给其他参与方。

(3) 对于 $i=2,\cdots,m$，第 i 方应用认证计算功能函数发送 r_i 给其他参与方。若第 i 方中

断或第 j 方没有获得期望输出，则第 j 方中断并输出⊥；否则，第 j 方得到认证计算功能函数的输出 r_i，且置 $r_i^j=r_i$，为简便起见，记 $r_j^j=r_j$。

(4) 若第一方未中断，则利用认证计算功能函数发送 $g(\bigoplus_{i=1}^{m} r_i^1)$ 给其他参与方。确切地讲，对于 $j=1,\cdots,m$，第 j 方令 $r^j=\bigoplus_{i=2}^{m} r_i^j$。第一方令 $r=r_1\oplus r^1$，以输入 (r_1,s_1,r^1)，第 j 方以输入 (c_1,r^j) 调用认证计算功能函数，其中第 j 方的输入与第一方输入关系应满足 $h(r_1,s_1,r^1)=(\bar{C}_{s_1}(r_1),r^1)$，第 j 方期望获得 $f(r_1,s_1,r^1)=g(r_1\oplus r^1)$。若第一方中断或者第 j 方从认证计算功能函数获得形如 $((\bar{C}_{s_1}(r_1),r^1),g(r_1\oplus r^1))$ 的结果，则第 j 方中断并输出⊥。

输出：第一方输出 r，第二方输出在第(4)步确定，或者是 $g(r)$ 或者是⊥。

第一类编译器也要用到输入承诺功能函数。m 方输入承诺功能函数：$(x,1^{|x|},\cdots,1^{|x|})\mapsto(r,\bar{C}_r(x),\cdots,\bar{C}_r(x))$，其中 r 在 $\{0,1\}^{|x|^2}$ 上均匀分布，将两方协议 6.9 中，对于认证计算功能函数和扩展掷币功能函数的调用，改为对于相应的多方功能函数的调用，即可安全实现输入承诺功能函数。

第一类编译器的工作方式是，给定一个半诚实模型中安全的 m 方协议 Π，生成第一类恶意模型中的 m 方协议 Π'，Π' 和 Π 完成同样的功能。具体实现方式是，将协议 6.10 构造方式中调用的功能函数替换成相应的多方功能函数，即得到第一类恶意模型下安全的 m 方协议 Π'。

6.4.5 第二类编译器

第二类编译器可以避免协议被中断执行，本小节的目的是将在第一类恶意模型中安全实现某个功能的协议，变换成在第二类恶意模型中安全实现同样功能的协议。当前所用的编译器与两方情形有很大不同，主要工具是可验证秘密共享方案，其详细讨论参见本书 7.7 节。下面的协议 6.17 是第二类编译器的构造。

协议 6.17(第二类多方编译器) 设 $t\leqslant m/2$，给定一个第一类恶意模型中的 m 方协议 Π，按以下方式产生第二类恶意模型中的 m 方协议 Π'。

输入：第 i 方的输入是 $x^i\in\{0,1\}^n$。

随机带：第 i 方均匀选择随机带 $r^i\in\{0,1\}^{c(n)}$。

分享阶段：每一方利用可验证秘密共享方案，将自己的输入和随机带与其他参与方分享。即对于 $i=1,\cdots,m$，第 i 方以输入 x^ir^i 调用可验证秘密共享方案，第 i 方从可验证秘密共享方案中获得的输出是 $(\bar{s}^i,\bar{\rho}^i)$，其他的如第 j 方获得 $(s_j^i,\rho_j^i,\bar{c}^i)$，其中 $\bar{s}^i=(s_1^i,\cdots,s_m^i)\leftarrow G_{m,t}(x^ir^i)$，$\bar{\rho}^i=(\rho_1^i,\cdots,\rho_m^i)$ 是均匀分布的，$\bar{c}^i=(c_1^i,\cdots,c_m^i)$，且 $c_k^i=\bar{C}_{\rho_k^i}(s_k^i)$。注意：这里所有诚实方获得正确的分享结果或者觉察第 i 方欺骗，若觉察出欺骗，则诚实方将分享结果设置为⊥。

中断处理：若第 i 方被觉察出欺骗，则诚实方将第 i 方的输入和随机带设置为某个值。

协议执行阶段：参与方利用认证计算功能函数模仿协议 Π 每一步的执行。假设当前步骤是第 i 方发送消息。调用认证计算功能函数的输入以及函数 h、f 定义如下。设 $\alpha=(\alpha_1,\alpha_2)$，$\alpha_1=(x^ir^i,\bar{s}^i,\bar{\rho}^i)$，$\alpha_2$ 是执行到当前步骤所发送的消息序列。$\beta_j=\beta=(\bar{c}^i,\alpha_2)$，$\alpha_2$ 如上所定义。函数 h 定义为 $h((z,(s_1,\cdots,s_m),(r_1,\cdots,r_m)),\gamma)=(\bar{C}_{r^1}(s_1),\cdots,\bar{C}_{r^m}(s_m),\gamma)$。函数 f 定义为当前步骤第 i 方要发送的消息。

中断处理：若某个参与方充当调用认证计算功能函数的发起方时中断，则其他参与方广播在分享阶段获得的他的输入和随机带的分享值，重构出输入及随机带。

输出：每个参与方持有即为此参与方在协议 Π 中的输出，直接输出。

6.5 小结

本章介绍的安全多方计算的主要结论来自于文献[2]和[3]。敌手模型是可以搭线窃听所有的通信信道，敌手攻击能力是非自适应的、恶意的，并且是计算有界的。对于其他类型的敌手也有重要的结果。例如，关于协议的公平性，自适应敌手以及协议并发安全性都有大量文献涉及。

在 6.2 节中曾指出，不存在完全公平的两方协议，即不能保证两方都获得输出结果。尽管如此，仍然可以放松对于完全公平性的要求，达到某种部分公平性，具体的定义和协议构造可参见文献[4]。自适应敌手比非自适应敌手的攻击能力更强，关于自适应模型安全性可参见文献[5]和[6]。

本章讨论的协议可以顺序复合，一个自然的问题是是否能够并发复合。关于协议并发复合安全性是当前国际上的热点研究问题，更一般的提法是协议是否能够普适复合，已有大量的结论，这方面的工作可参见文献[7]～[9]。

本章在写作过程中得到了徐海霞副教授的大力支持，她提供了大量的相关材料及她的一些理解，作者在此表示衷心的感谢。

参 考 文 献

[1] Oded Goldreich. Foundations of cryptography. Volume II, Basic Applications.

[2] Yao A C. How to generate and exchange secrets. In 27th IEEE FOCS, 1986,162～167.

[3] Goldreich O, Micali S, Wigderson A. How to play any mental game-A completeness theorem for protocols with honest majority. In 19th ACM STOC, 1987,218～229.

[4] Cleve R. Limits on the security of coin flips when half the professors are faulty. In 18th ACM STOC, 1986,364～369.

[5] Canetti R, Damgard I, Dziembowski S. On adaptive versus non-adaptive security of multiparty protocols. Journal of Crptology, Vol. 17. No. 3:153～207,2004.

[6] Canetti R, Feige U, Goldreich O. Adaptively secure multiparty computation. In 28th ACM STOC, 1996,639～648.

[7] Canetti R. Security and composition of multiparty cryptographic protocols. Journal of Crptology, Vol. 13, No. 1,2000,143～202.

[8] Canetti R. Universally composable Security: A new paradigm for cryptographic protocols. In 42nd IEEE FOCS, 2001,136～145.

[9] Canetti R, Lindell Y, Ostrovsky R. Universally composable two-party and multi-party secure computation. In 34th ACM, STOC, 2002,494～503.

第3篇　基础安全协议

基础安全协议主要是指与具体实际应用无关的一类基础协议，是设计应用安全协议或其他大型安全协议的基础。本篇主要介绍秘密共享、数字签名、身份识别、密钥交换、健忘传输、公平交换等几大类基础安全协议以及一些常用的攻击基础安全协议的方法。

第 7 章　秘密共享协议

秘密共享协议本质上是一种简单的安全多方计算协议，是构建各类安全协议的基础模块，如利用秘密共享协议可设计出各种类型的门限数字签名协议。特别是秘密共享的基本思想在信息系统安全中有着极其重要的应用价值，如利用这种思想或技术可增强数字认证系统 CA 和数据库系统的安全性。

目前已利用各种方法诸如代数方法、组合方法和几何方法构造出了大量的适应于各种不同环境的秘密共享协议，如基本的秘密共享协议、黑箱秘密共享协议、多重秘密共享协议、可验证的秘密共享协议及前摄秘密共享协议。也有很多讨论秘密共享协议的有效性和结构等的研究论文。本章主要介绍几类典型的秘密共享协议。

7.1　秘密共享的基本思想

先用一个例子来说明秘密共享的基本思想和观点。例如，存储在系统中的所有密钥的安全性(从而整个系统的安全性)可能最终取决于一个主密钥。但这样做有两个缺陷：一是若主密钥偶然地或蓄意地被暴露，整个系统就易受攻击；二是若主密钥丢失或毁坏，系统中的所有信息就用不成了。后一个问题可通过将密钥的副本发给信得过的用户来解决。但这样做的话，系统对背叛行为又无法对付。解决这两个问题的一个办法就是使用秘密共享协议。

秘密共享协议的基本思想是将密钥 k 按下述方式分成 n 个分享 $k_1,k_2,\cdots,k_n$：

(1) 已知任意 t 个 k_i 值易于计算出 k。

(2) 已知任意 $t-1$ 个或更少个 k_i，则由于信息短缺而不能计算出 k。

这种方法也称为(t,n)门限法，对应的序列 $k_1,k_2,\cdots,k_n$ 称为(t,n)门限序列。

将 n 个分享 $k_1,k_2,\cdots,k_n$ 分给 n 个用户。由于重构密钥至少需要 t 个分享，故暴露 $s(s\leqslant t-1)$个分享不会危及密钥，从而少于 t 个用户的共谋不能得到密钥。同时，若一个分享被丢失或毁坏，仍可恢复密钥(只要至少有 t 个有效的分享)。这种方法也可用于保护任何类型的数据。

7.2　基本的秘密共享协议

本节介绍两个最基本的秘密共享协议，即 Shamir 秘密共享协议和 Asmuth-Bloom 秘密共享协议。基本的秘密共享协议是其他秘密共享协议的核心基础，也是构造其他安全协议或安全方案的基本工具。

7.2.1　Shamir 秘密共享协议

Shamir于1979年基于拉格朗日插值公式提出了一个秘密共享协议[1]，也称为

门限方案。

首先介绍一下拉格朗日插值公式。设 p 是素数，$x_1, x_2, \cdots, x_t$ 是 Z_p 中不同的元素，设 $y_1, y_2, \cdots, y_t$ 是 Z_p 中的元素（未必不同）。则存在次数至多为 $t-1$ 的唯一的多项式 $h(x) \in Z_p[x]$，使得 $h(x_i) = y_i, 1 \leqslant i \leqslant t$。并且多项式 $h(x)$ 为

$$h(x) = \sum_{s=1}^{t} y_s \prod_{\substack{j=1 \\ j \neq s}}^{t} \frac{x - x_j}{x_s - x_j}$$

上述公式很容易证明，留给读者作为练习自行完成。

假定 p 是一个素数，共享的秘密（也称为共享秘密）$k \in K = Z_p$。可信中心 TA 给 $n(n<p)$ 个分享者 $P_i(1 \leqslant i \leqslant n)$ 分配分享的过程如下。

(1) TA 随机选择一个 $t-1$ 次多项式 $h(x) = a_{t-1}x^{t-1} + \cdots + a_1 x + a_0 \in Z_p[x]$，常数 $a_0 = k$。

(2) TA 在 Z_p 中选择 n 个非零的、互不相同的元素 $x_1, x_2, \cdots, x_n$，计算 $y_i = h_i(x_i)$，$1 \leqslant i \leqslant n$。

(3) TA 将 $(x_i, y_i)(1 \leqslant i \leqslant n)$ 分配给分享者 $P_i(1 \leqslant i \leqslant n)$，值 x_i 是公开知道的，y_i 作为 P_i 的秘密分享。

每个数对 (x_i, y_i) 是"曲线"$h(x)$ 上的一个点。因为 t 个点唯一地确定 $t-1$ 次多项式 $h(x)$，所以 k 可以从 t 个分享重构出。但是从 $t_1(t_1<t)$ 个分享无法确定 $h(x)$，从而无法确定 k。

给定 t 个分享 $y_{i_s}(1 \leqslant s \leqslant t)$，从拉格朗日插值公式重构的 $h(x)$ 为

$$h(x) = \sum_{s=1}^{t} y_{i_s} \prod_{\substack{j=1 \\ j \neq s}}^{t} \frac{x - x_{i_j}}{x_{i_s} - x_{i_j}}$$

运算都是 Z_p 上的运算即模 p 运算。

一旦知道 $h(x)$，通过 $k = h(0)$ 易于计算出秘密 k。因为 $k = h(0) = \sum_{s=1}^{t} y_{i_s} \prod_{\substack{j=1 \\ j \neq s}}^{t} \frac{-x_{i_j}}{x_{i_s} - x_{i_j}}$，若令 $b_s = \prod_{\substack{j=1 \\ j \neq s}}^{t} \frac{-x_{i_j}}{x_{i_s} - x_{i_j}}$，则 $k = h(0) = \sum_{s=1}^{t} b_s y_{i_s}$。因为 $x_i(1 \leqslant i \leqslant n)$ 的值是公开知道的，所以可预计算 $b_s(1 \leqslant s \leqslant n)$ 以加快重构秘密 k 的运算速度。

7.2.2 Asmuth-Bloom 秘密共享协议

Asmuth 和 Bloom 于 1980 年基于中国剩余定理提出了一个秘密共享协议[2]。在他们的协议中，分享是和共享秘密 k 相联系的一个数的某些同余类。令 $p, d_1, \cdots, d_n$ 是协议中公开的一组数，它们满足下列条件：

(1) $p > k$。

(2) $d_1 < d_2 < \cdots < d_n$。

(3) 对所有的 i，$\gcd(p, d_i) = 1$；对 $i \neq j$，$\gcd(d_i, d_j) = 1$。

(4) $d_1 d_2 \cdots d_t > p d_{n-t+2} d_{n-t+3} \cdots d_n$。

令 $N = d_1 d_2 \cdots d_t$，N/p 大于任意 $t-1$ 个 d_i 之积，r 是区域$[0, N/p-1]$中的一个随机整数。计算 $k' = k + rp$，$k' \in [0, N-1]$。n 个分享为：

$k_i=k'\bmod d_i, i=1,2,\cdots,n$,共享秘密 k 可以由任意 t 个分享重新恢复。

为了恢复 k,找到 k'就足够了,因为从 k'、r、p,可以计算 $k=k'-rp$。若给定 t 个分享 $k_{i_1},\cdots,k_{i_t}$,则由中国剩余定理知,同余方程组

$$k'\equiv k_{i_1}(\bmod d_{i_1})$$
$$k'\equiv k_{i_2}(\bmod d_{i_2})$$
$$\cdots$$
$$k'\equiv k_{i_t}(\bmod d_{i_t})$$

在$[0,N_1-1]$内有唯一解,将它记作 x,这里 $N_1=d_{i_1}d_{i_2}\cdots d_{i_t}$。因为 $N_1\geqslant N$,这就唯一地确定了 k',$k'=x \bmod N, 0\leqslant k'<N$。

若仅知道 $t-1$ 个分享 $k_{i_1},k_{i_2},\cdots,k_{i_{t-1}}$,只能确定 $k'\equiv x \bmod N_2$,这里 $0\leqslant x<N_2=d_{i_1}d_{i_2}\cdots d_{i_{t-1}}$,$x$ 是 $t-1$ 个分享确定的同余方程的解。由此只能确定 k'的取值集合是 $S_p=\{x+N_2j:0\leqslant j<N/N_2\}$。注意 $N/N_2>p$,$\gcd(p,N_2)=1$,故 S_p 中的元素模 p 分布是均匀的。

7.3 一般存取结构上的秘密共享协议

在 7.2 节介绍的基本秘密共享协议中,n 个参与者中的任意 k 个都可以恢复秘密,因此 n 个参与者的权限是相同的。但在实际应用中,参与者的职位和分工不同,其权限往往也是不同的,并不是任意的某些人而是符合特定要求的一些人组合在一起才可以恢复秘密,因此需要考虑一般情况下的秘密共享协议。Ito 等人于 1987 年提出了存取结构的概念[3],并给出了一般存取结构上的秘密共享协议。本节主要介绍 Ito 等人构造的一般存取结构上的秘密共享协议。

一般地,在一个秘密共享协议中,设秘密为 s,参与者(也称参与方、分享者,本章中交替使用了这些术语)集合为 $P=\{P_1,\cdots,P_n\}$,2^P 为 P 的幂集。分发者 D(充当可信中心的角色)利用秘密 s 生成 n 个秘密分享,分别为 $y_1,\cdots,y_n$,然后将 y_i 分发给 P_i。设 $P'=\{P_{i_1},\cdots,P_{i_t}\}\subseteq P$,如果 P'利用$\{y_{i_1},\cdots,y_{i_t}\}$能够恢复秘密 s,则称 P'为一个授权子集,否则,称为一个非授权子集。所有授权子集组成的集合称为该秘密共享协议的存取结构,记为 Γ,同时称 $\Omega=2^P-\Gamma$ 为敌手结构。

如果 P'为授权子集,则任意集合 $P''(P'\subseteq P''\subseteq P)$也是授权子集,因此存取结构 Γ 满足单调性(单调上升),即

$$A\in\Gamma \quad 且 \quad A\subseteq A'\subseteq P, \quad 则 \quad A'\in\Gamma \tag{7-1}$$

由于 Γ 满足单调性,因此 $\Omega=2^P-\Gamma$ 也满足单调性(单调递减),即

$$B\in\Omega \quad 且 \quad B'\subseteq B\subseteq P, \quad 则 \quad B'\in\Omega \tag{7-2}$$

对于任意的存取结构 Γ,定义极小授权子集 $A\in\Gamma$ 为满足对任意的 $A'\subset A$,$A'\notin\Gamma$ 的集合。Γ 的基(Basis)Γ_0 定义为所有极小授权子集组成的集合。Γ_0 可以唯一确定 Γ,反之亦然。因此,在描述 Γ 时,只需给出 Γ_0 即可。

类似地,定义极大非授权子集 $B\in\Omega$ 为满足对任意的 $B'\supset B$,$B'\notin\Omega$ 的集合,极大敌手结构 Ω_M为所有极大非授权子集组成的集合。

文献[3]中指出,任意满足式(7-1)的参与者集合的子集族都可以作为存取结构。下面

介绍文献[3]中构造的一般存取结构上的秘密共享协议,也称为多分配方案(Multiple Assignment Scheme)。

设参与者集合为 $P=\{P_1,\cdots,P_n\}$,现有满足式(7-1)的子集族 $\Gamma\subseteq 2^P$,将构造以 Γ 为存取结构的秘密共享协议。Ω_M为极大敌手结构,其元素个数为$|\Omega_M|=k$。方便起见,不妨将Ω_M记作 $\Omega_M=\{B_1,B_2,\cdots,B_k\}$。具体协议如下。

(1) 选择素数 p,$k\leqslant n<p$,构造有限域 $F_p=Z_p$。

(2) 随机选择 $a_0,a_1,\cdots,a_{k-1}\in F_p$,令 $a_0=s$。构造 $k-1$ 次多项式 $f(x)=a_0+a_1x+a_2x^2+\cdots+a_{k-1}x^{k-1}$。

(3) 随机选择 k 个非零的、不同元素 $x_1,x_2,\cdots,x_k\in F_p$,计算 $y_i=f(x_i)$,$i=1,\cdots,k$。

(4) 设 $S=\{(x_i,y_i):B_i\in\Omega_M,1\leqslant i\leqslant k\}$,$\Omega_M$ 与 S 存在一一对应的关系,Ω_M 的每一个元素都对应着 S 中的一个二元组。

(5) 定义映射

$g:P\rightarrow 2^S$,使得 $S_i=g(P_i)=\{(x_j,y_j):1\leqslant j\leqslant k,P_i\notin B_j\in\Omega_M\},1\leqslant i\leqslant n$

分发者按照映射 g 将 S_i 分发给 $P_i(1\leqslant i\leqslant n)$,作为 P_i 的秘密分享。现在来说明上述由 g、S 所构造的秘密共享协议的存取结构是 Γ。

首先说明授权集 A 能够恢复秘密 s。

因为 $\Omega_M\subseteq\Omega=2^P-\Gamma$,所以对每个 $B_j\in\Omega_M$,肯定存在 $P_i\in A$,使得 $P_i\notin B_j$,$1\leqslant i\leqslant n$,$1\leqslant j\leqslant k$,因此 A 拥有的秘密分享满足 $|\bigcup_{P_i\in A}S_i|=k$,按照 7.2.1 节介绍的秘密恢复方法,$A$ 可以恢复秘密 s。

其次说明当 $A\notin\Gamma$ 时,A 无法确定或恢复秘密 s。

由于 $A\notin\Gamma$,则 $A\in\Omega$。由 Ω_M的定义,存在 $B_j\in\Omega_M$,使 $A\subseteq B_j$。再由 S_i 的定义得,$(x_j,y_j)\notin\bigcup_{P_i\in A}S_i$,即 A 拥有的秘密分享数至多为 $k-1$,因此 A 无法确定或恢复秘密 s。

综上所述,由 g、S 所构造的秘密共享协议的存取结构确实是 Γ。

上述协议表明,对于 2^P 上任何满足式(7-1)的集合族,都可以作为一个存取结构,存在一个实现这个存取结构的秘密共享协议,因而一般存取结构的概念是有实际意义的。

7.4 黑箱秘密共享协议

通常的秘密共享协议都是建立在特定的秘密空间之上,设计协议时通常是事先已经给定了相应的秘密空间,然后按照存取结构进行设计。但是在某些秘密共享协议的应用环境中,有可能并不知道秘密空间的相关信息或者了解到的信息非常有限。例如,在基于群的分布式密码系统中,并不知道群的阶,而仅仅知道它是一个有限群。为此,Cramer 和 Fehr 提出了黑箱秘密共享协议来解决这一问题[4]。本节主要介绍 Cramer 和 Fehr 构造的黑箱秘密共享协议。

简单地讲,黑箱秘密共享协议与通常的秘密共享协议的不同之处在于:黑箱秘密共享协议共享的秘密空间为任意的有限交换群 G,群 G 中元素之间的运算以及秘密分享生成过程中随机元素的选取都以黑箱方式调用。秘密和秘密分享都取值于 G,而协议的分发矩阵(Distribution Matrix)和重构向量(Reconstruction Vectors)都定义在整数环$\mathbb{Z}$ 上,且与 G 独

立无关。

首先给出黑箱秘密共享协议的一个基本定义。

设 $P=\{P_1,\cdots,P_n\}$ 为参与者集合，Γ 为 P 上的存取结构。R 为含单位元的交换环，矩阵 $\boldsymbol{M}\in R^{d,e}$，映射 $\varphi:\{1,\cdots,d\}\to\{P_1,\cdots,P_n\}$ 为满射。称 $\boldsymbol{M}$ 的第 j 行标号为 $\varphi(j)$ 或者"$\varphi(j)$ 拥有第 j 行"。对于 $A\subset P$，M_A 为 $\boldsymbol{M}$ 中 A 拥有的行，d_A 为 M_A 的行数。对于任意 $A\in\Gamma$，设 $\boldsymbol{\lambda}(A)\in R^{d_A}$ 为一列向量，称为 A 的重构向量，$\mathfrak{R}=\{\boldsymbol{\lambda}(A)\mid A\in\Gamma\}$。$\boldsymbol{\varepsilon}=(1,0,\cdots,0)^{\mathrm{T}}\in R^e$ 称为目标向量。$M=(R,\boldsymbol{M},\varphi,\boldsymbol{\varepsilon})$ 称为(环 R 上的)单调张成方案，简称为 MSP(Monotone Span Program)。如果 $R=\mathbb{Z}$，则称 M 为整数张成方案，简称为 ISP(Integer Span Program)。

称 $M=(\mathbb{Z},\boldsymbol{M},\varphi,\boldsymbol{\varepsilon})$ 是 Γ 上的黑箱秘密共享协议，如果下列条件成立：设 G 为任意的 Abel 群(这里以加法群为例)，$A\subset P$ 为任意的非空集合。对于任意分布的秘密 $s\in G$，设 $\boldsymbol{g}=(g_1,\cdots,g_e)^{\mathrm{T}}$ 均匀分布于 G^e，$g_1=s$。定义 $\boldsymbol{s}=\boldsymbol{M}\boldsymbol{g}$。则

(1) 完全性。如果 $A\in\Gamma$，则 $\boldsymbol{M}_A^{\mathrm{T}}\boldsymbol{\lambda}(A)=\boldsymbol{\varepsilon}$。即恢复秘密时 $\boldsymbol{s}_A^{\mathrm{T}}\boldsymbol{\lambda}(A)=(\boldsymbol{M}_A\boldsymbol{g})^{\mathrm{T}}\boldsymbol{\lambda}(A)=\boldsymbol{g}^{\mathrm{T}}\boldsymbol{M}_A^{\mathrm{T}}\boldsymbol{\lambda}(A)=s$ 以概率 1 成立，其中 $\boldsymbol{\lambda}(A)\in\mathfrak{R}$ 为 A 的重构向量。

(2) 秘密性。如果 $A\notin\Gamma$，则存在 $\boldsymbol{\kappa}=(\kappa_1,\cdots,\kappa_e)^{\mathrm{T}}\in\ker\boldsymbol{M}_A$，其中 $\kappa_1=1$。也就是说，要保证 $\boldsymbol{s}_A$ 不含 s 的任何信息，即 $H(s\mid\boldsymbol{s}_A)=0$。

引理 7.1　设 $f(X)\in\mathbb{Z}[X]$ 为一首 1 不可约多项式，$m=\deg(f)$。考虑环 $R=\mathbb{Z}[X]/(f(X))$，设 $M=(R,\boldsymbol{M},\varphi,\boldsymbol{\varepsilon})$ 为定义在 R 上的 Γ 上的 MSP，则存在 Γ 上的 ISP $\hat{M}=(\mathbb{Z},\hat{\boldsymbol{M}},\hat{\varphi},\hat{\boldsymbol{\varepsilon}})$ 且 $\mathrm{size}(\hat{M})=m\cdot\mathrm{size}(M)$。

引理 7.1 的具体证明过程以及 Γ 上的 MSP 的定义可参阅文献[4]。

由引理 7.1 可知，构造 Γ 上的黑箱秘密共享协议只需构造出 Γ 上的 MSP 即可。

下面简要介绍 Cramer 和 Fehr 给出的门限存取结构 $\Gamma_{t,n}$ 上的一个黑箱秘密共享协议，其中 $\Gamma_{t,n}=\{A\subset P\mid |A|\geqslant t\}$。

设 G 为加法交换群，秘密 s 均取自于 G。对于 $t=1$ 和 $t=n$ 这两种平凡情况，可简单地做以下处理。

(1) 当 $t=1$ 时，P 中的任意一个人都可以独自恢复秘密 s，这时直接将 s 作为每个人的秘密分享进行分发即可。

(2) 当 $t=n$ 时，P 中所有的 n 个人联合才可以恢复秘密 s，此时随机选取 $y_1,\cdots,y_n\in G$，使得 $\sum_{i=1}^{n}y_i=s$，将 y_i 分发给 $P_i(i=1,\cdots,n)$，作为他的秘密分享即可。

下面考虑 $1<t<n$ 的情况。

设 R 为含单位元的交换环，首先来构造定义在 R 上的门限存取结构 $T_{t,n}$ 对应的 MSP，进而找出 R 所需具备的性质，然后再构造满足该性质的环 R。

对任意的 $x_1,x_2,\cdots,x_n\in R$，定义

$$\Delta(x_1,\cdots,x_n)=\prod_{i=1}^{n}x_i\cdot\prod_{1\leqslant j<i\leqslant n}(x_i-x_j)=\begin{vmatrix}x_1 & x_1^2 & \cdots & x_1^n\\ x_2 & x_2^2 & \cdots & x_2^n\\ \vdots & \vdots & \ddots & \vdots\\ x_n & x_n^2 & \cdots & x_n^n\end{vmatrix}$$

假设存在 $\alpha_1,\cdots,\alpha_n\in R$ 以及 $r_0,r_1\in R$，使得

$$r_0\cdot\Delta(1,\cdots,n)^2+r_1\cdot\Delta(\alpha_1,\cdots,\alpha_n)^2=1$$

令 $\Delta_0=\Delta(1,\cdots,n)\in R$，$\Delta_1=\Delta(\alpha_1,\cdots,\alpha_n)\in R$，构造以下矩阵：

$$\boldsymbol{N}_0=\begin{bmatrix}\Delta_0 & 1 & 1^2 & \cdots & 1^{t-1}\\ \Delta_0 & 2 & 2^2 & \cdots & 2^{t-1}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \Delta_0 & n & n^2 & \cdots & n^{t-1}\end{bmatrix}\in R^{n,t},$$

$$\boldsymbol{N}_1=\begin{bmatrix}\Delta_1 & \alpha_1 & \alpha_1^2 & \cdots & \alpha_1^{t-1}\\ \Delta_1 & \alpha_2 & \alpha_2^2 & \cdots & \alpha_2^{t-1}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \Delta_1 & \alpha_n & \alpha_n^2 & \cdots & \alpha_n^{t-1}\end{bmatrix}\in R^{n,t}$$

$\boldsymbol{N}_0$、$\boldsymbol{N}_1$ 第 i 行的标号为 P_i。

对任意的 $A\notin T_{t,n}$，$\boldsymbol{N}_0$，$\boldsymbol{N}_1$ 均存在 $\boldsymbol{\kappa}_b=(\kappa_{b1},\cdots,\kappa_{bt})^{\mathrm{T}}\in\ker N_{bA}$，其中 $\kappa_{b1}=1$（$N_{bA}(b=0,1)$ 表示 A 在 $\boldsymbol{N}_b$ 中所对应的行）。而对任意的 $A\in T_{t,n}$，$(\Delta_0^2,0,\cdots,0)\in\operatorname{span}(N_{0A})\in R^t$，$(\Delta_1^2,0,\cdots,0)\in\operatorname{span}(N_{1A})\in R^t$。根据 Cramer 法则和 Vandermonde 行列式的性质容易验证上面的结论。

在 $\boldsymbol{N}_0$、$\boldsymbol{N}_1$ 的基础上，定义一个新的单调张成方案的矩阵 $\boldsymbol{M}\in R^{2n,2t-1}$，

$$\boldsymbol{M}=\begin{bmatrix}\Delta_0 & 1 & \cdots & 1^{t-1} & 0 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots\\ \Delta_0 & n & \cdots & n^{t-1} & 0 & \cdots & 0\\ \Delta_1 & 0 & \cdots & 0 & \alpha_1 & \cdots & \alpha_1^{t-1}\\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots\\ \Delta_1 & 0 & \cdots & 0 & \alpha_n & \cdots & \alpha_n^{t-1}\end{bmatrix}\begin{matrix}P_1\\ \vdots\\ P_n\\ P_1\\ \vdots\\ P_n\end{matrix}$$

每行中 0 的个数为 $t-1$，每行标号与 $\boldsymbol{N}_1$、$\boldsymbol{N}_0$ 保持一致。定义目标向量 $\boldsymbol{\varepsilon}=(1,0,\cdots,0)^{\mathrm{T}}\in R^{2t-1}$，由上面 $\boldsymbol{N}_1$、$\boldsymbol{N}_0$ 的性质可知，当 $A\notin T_{t,n}$ 时，存在 $\boldsymbol{\kappa}=(\kappa_1,\cdots,\kappa_{2t-1})^{\mathrm{T}}\in\ker M_A$，其中 $\kappa_1=1$；而当 $A\in T_{t,n}$ 时，则 M_A 的各行可以扩展得到目标向量：它们可以扩展得到向量 $(r\cdot\Delta_0^2+s\cdot\Delta_1^2,0,\cdots,0)\in R^{2t-1}$，其中 $r,s\in R$，令 $r=r_0$，$s=r_1$，则由 M_A 的各行扩展得到了目标向量 $\boldsymbol{\varepsilon}$。

因此，上述构造的单调张成方案能够计算 $T_{t,n}$。

下面来构造 R，它是 $\mathbb{Z}$ 上的 $O(\log_2 n)$ 次扩张，且 $\alpha_1,\cdots,\alpha_n,r_0,r_1\in R$，$r_0\cdot\Delta_0^2+r_1\cdot\Delta_1^2=1$，其中 $\Delta_0=\Delta(1,\cdots,n)$，$\Delta_1=\Delta(\alpha_1,\cdots,\alpha_n)$。

设 $\Pi_n=\{p\mid 2\leqslant p\leqslant n, p\text{ 为素数}\}$，定义 $Q_n=\prod\limits_{p\in\Pi_n}p\in\mathbb{Z}$。令 $m=\lfloor\log_2 n\rfloor+1$，设 $\hat{f}(X)\in\mathbb{Z}[X]$ 为任意的首一 m 次不可约多项式，使得对于任意 $p\in\Pi_n$，$\hat{f}_p(X)$（$\hat{f}(X)$ 的系数 $\bmod p$）在 $\mathbb{F}_p[X]$ 中均不可约。可利用中国剩余定理求出 $\hat{f}(X)$。

设 $R=\mathbb{Z}[X]/(\hat{f}(X))$。由 $\hat{f}(X)$ 的定义可得，对于任意 $p\in\Pi_n$，$R/(p)$ 为有限域。事实上，对于任意 $p\in\Pi_n$，有

$$R/(p) \simeq \mathbb{Z}[X]/(p, \hat{f}(X)) \simeq \mathbb{F}_p[X]/(\hat{f}_p(X)) \simeq \mathbb{F}_{p^m}$$

注意到 R 中的所有理想 (p) 互异且都是极大理想，$p \in \Pi_n$，利用中国剩余定理可得

$$R/(Q_n) \simeq \prod_{p \in \Pi_n} \mathbb{F}_{p^m}。$$

对于任意 $p \in \Pi_n$，$|\mathbb{F}_{p^m}^*| = p^m - 1 \geqslant 2^m - 1 \geqslant n$，因此存在非零的 $\beta_1^{(p)}, \cdots, \beta_n^{(p)} \in \mathbb{F}_{p^m}$。

任选 $\alpha_1, \cdots, \alpha_n \in R$，使得 $R/(Q_n) \ni \bar{\alpha}_i \leftrightarrow (\beta_i^{(p)})_{p \in \Pi_n} \in \prod_{p \in \Pi_n} \mathbb{F}_{p^m}$，$i = 1, \cdots, n$。于是对于任意的 i, j，$1 \leqslant i \neq j \leqslant n$ 有 $\bar{\alpha}_i \in (R/(Q_n))^*$（注：$(R/(Q_n))^*$ 为 $R/(Q_n)$ 的乘法群）以及 $\bar{\alpha}_i - \bar{\alpha}_j \in (R/(Q_n))^*$，因此 $\bar{\Delta}_1 \in (R/(Q_n))^*$，则对于任意的正整数 k，都有 $\bar{\Delta}_1 \in (R/(Q_n^k))^*$，可利用递归来验证此结论。假设存在 $v, w \in R$，使得 $\Delta_1 \cdot v = 1 + w \cdot Q_n^i$，$i \geqslant 1$，则 $\Delta_1 \cdot (v - vw \cdot Q_n^i) = 1 - w^2 \cdot Q_n^{2i}$，$2i \geqslant i+1$，从而 $\bar{\Delta}_1 \in (R/(\Delta_0^2))^*$。亦即，作为一个整数，$\Delta_0^2$ 在素数 $p \in \Pi_n$ 上可以完全分解。选择足够大的 k^*，使得 Δ_0^2 整除 $Q_n^{k^*}$，由前面的讨论可知 $\bar{\Delta}_1^2 \in (R/(\Delta_0^2))^*$，因此存在 $r_0, r_1 \in R$，使得 $r_0 \cdot \Delta_0^2 + r_1 \Delta_1^2 = 1$。

由引理 7.1 及其证明，可以得出下列结论。

引理 7.2　对于所有的 t, n，$2 \leqslant t < n$，存在 $\Gamma_{t,n}$ 上的大小为 $n \cdot (\lfloor \log_2 n \rfloor + 2)$ 的 ISP。

综上所述，可以得出上述构造的 $\Gamma_{t,n}$ 上的秘密共享协议确实是一个黑箱秘密共享协议。

实际上，该协议是把秘密分发了两次，一次是以 $1, \cdots, n$ 为插值点，另一次是以 $\alpha_1, \cdots, \alpha_n$ 为插值点，使得 $\Delta_0 = \Delta(1, \cdots, n)$，$\Delta_1 = \Delta(\alpha_1, \cdots, \alpha_n)$ 互素，对于任何一个授权集合来说，都可以恢复得到 $(\Delta_0^2, 0, \cdots, 0) \in R^{2t-1}$ 和 $(\Delta_1^2, 0, \cdots, 0) \in R^{2t-1}$，进而可以找到 r_0、r_1，使得 $(r_0 \cdot \Delta_0^2 + r_1 \cdot \Delta_1^2, 0, \cdots, 0) = (1, 0, \cdots, 0) \in R^{2t-1}$，进而可以恢复出秘密。

Cramer 等人后来的工作给出了 $\Gamma_{t,n}$ 门限存取结构上的另外一种更优的黑箱秘密共享协议，感兴趣的读者可参阅文献[5]。

7.5　无限取值空间上的秘密共享协议

前面介绍的秘密共享协议中的秘密和秘密分享的取值空间都是有限的，那么对于取值空间是无限的情况来说，其上的秘密共享协议是否还存在呢？Chor 和 Kushilevitz 研究了这个问题[6]，特别是秘密和秘密分享都取自于一个可数集合（如所有的二进制串）的情况。他们证明了任意可数取值空间上的秘密共享协议（参与者人数 $n \geqslant 2$）是不存在的，而实数域上的秘密共享协议是存在的，这说明秘密共享协议的存在性与域的基数有关。本节主要介绍 Chor 和 Kushilevitz 的结果。

设 $P = \{P_1, P_2, \cdots, P_n\}$，$\Gamma$ 为 P 上的存取结构。如果存在集合 $A \in \Gamma$，满足 A 不是一个单元集，且是 Γ 中最小的一个，则称 Γ 是**非平凡的**。

设 S 为可能秘密的任意集合，Γ 是一个非平凡的存取结构，设 $\alpha \geqslant 1$ 为一常数。S 上的 (Γ, α) 秘密共享协议是一个由秘密集合到 n 元组集合的概率映射 $\Pi: S \to B_1 \times B_2 \times \cdots \times B_n$，并满足以下条件：

(1) 秘密 s 能够由任意的“合法的”秘密分享集合恢复。也就是说，对于任意的子集 $A \in \Gamma$，存在函数 $h_A: \times_{i \in A} B_i \to S$，使得对于每个可能的秘密分享集合 $(s_1, \cdots, s_n) = \Pi(s)$，秘密都可以由 $h_A(\{s_i\}_{i \in A}) = s$ 恢复。

(2)“非法的”秘密分享集合不会泄露“过多”的关于秘密的信息(信息论意义上的)。即对于任意的子集 $A\notin\Gamma$,任意的两个秘密 $s,s'\in S$ 及任意可能的秘密分享集 $\{s_i\}_{i\in A}$:

$$\frac{1}{\alpha}\cdot\Pr(\{s_i\}_{i\in A}\mid s')\leqslant\Pr(\{s_i\}_{i\in A}\mid s)\leqslant\alpha\cdot\Pr(\{s_i\}_{i\in A}\mid s')$$

当 $\alpha=1$ 时,称此时的 (Γ,α) 秘密共享协议是完美的。在一个完美的秘密共享协议中,任何“非法的”秘密分享都不会泄露关于秘密的任何信息。

显然,如果秘密的取值空间是无限的,那么秘密分享的取值空间也一定是无限的。如果考虑秘密的取值空间是可数的秘密共享协议,那么希望秘密分享的取值空间最好也是可数的。下面的定理 7.1 说明了秘密和秘密分享都取自可数集合的秘密共享协议是不存在的。

定理 7.1 设 A 是一个可数集合,$n\geqslant2,\alpha\geqslant1,\Gamma$ 是一个非平凡的存取结构。则不存在一个 (Γ,α) 秘密共享协议,使得分发的秘密取自于 A,而秘密分享取自一个可数集合。

定理 7.1 的详细证明可参阅文献[6]。

虽然实数域上的秘密共享协议并没有什么实际的应用,但是对于探讨是否任意的无限集合上都不存在秘密共享协议这个问题来说,却很有帮助。下面介绍实数域上的一个简单的秘密共享协议。由于实数和单位区间[0,1)之间存在着一一对应关系,为简便起见,将此区间作为秘密的取值集合,而秘密分享同样取值于此集合。

首先定义什么是实数域上的秘密共享协议,特别是要给出非授权集(即 $A\notin\Gamma$)不会泄露关于秘密的任何信息的确切定义(这里介绍的所有协议都是完美的($\alpha=1$),故将 α 省略)。具体定义为:对于任意的两个秘密 $s,s'\in S$,任意的 $A\notin\Gamma$ 以及任意的 $|A|$ 元组可测集 $\{C_i\}_{i\in A}\subseteq[0,1)$,下列条件成立:

$$\Pr(\forall i\in A:s_i\in C_i\mid s)=\Pr(\forall i\in A:s_i\in C_i\mid s')$$

现在介绍任意非平凡存取结构 $\Gamma(n\geqslant2)$ 上的秘密共享协议。首先介绍一个 (k,k) 秘密共享协议,秘密 s 取自[0,1)区间。在此使用[0,1)上的 Legesgeu 测度。

(1) 按照均匀分布,独立地选择[0,1)区间上的 $k-1$ 个实数 $s_1,\cdots,s_{k-1}$。

(2) 选择 $s_k\in[0,1)$,满足 $s_1+\cdots+s_{k-1}+s_k=s(\text{mod }1)$。

证明上述协议是一个秘密共享协议的方法与素域上的情况类似。

对于任意一个非平凡的存取结构 Γ,其上的秘密共享协议可以利用文献[7]中的方法得到。

推理 7.1 设 $\Gamma(n\geqslant2)$ 是一个非平凡的存取结构,存在一个完美的 Γ 秘密共享协议,其共享的秘密取自于一个可数集合,生成的秘密分享均为实数。

可以将取自于可数集合的秘密嵌入到[0,1)区间上,按照上面的协议进行共享。易见其结果确实为一个秘密共享协议。

可数集合上的秘密共享与实数域上的秘密共享的不同之处就在于两者基数 $\aleph_0$ 和 $\aleph$ 的不同。

由上面的讨论可以看出,秘密共享协议的存在与否不单单要看取值空间是有限还是无限,还要看空间的基数。Chor 和 Kushilevitz[6] 给出了以下的结论。

(1) 如果秘密和秘密分享都取自可数空间(即其与正整数集合的基数相同),则对于任意的 $n\geqslant2$、一个非平凡的存取结构 Γ,以及 $\alpha\geqslant1$,(Γ,α) 秘密共享协议是不存在的。

(2) 如果秘密和秘密分享的取值空间的基数为 $\aleph$(实数域的基数),则对于任意一个非

平凡的存取结构 Γ，Γ 完美秘密共享协议是存在的。

Blakley 和 Swanson 也给出了相似的结果[8]，但 Chor 和 Kushilevitz 的结论更一般化，适用于一般的 α 弱秘密共享协议，而文献[8]中的结论仅适用于完美的门限方案，而且 Chor 和 Kushilevitz 的结论的条件和证明更加简单一些。

7.6　在线秘密共享协议

前面所介绍的秘密共享协议在秘密恢复之前，参与者集合是保持不变的。而在实际应用中，人员的变动是难免的，如企业中的人员招聘、内部人员调动、升迁、离退休等都会对企业原有信息系统造成一定的影响。因此，希望有一种灵活且易于实现的秘密共享协议来解决该问题。Blakley 等人[9]研究了带除名(Disenrollment)的门限方案，参与者可以自由离去并公开他的秘密分享，然后分发者 D(Dealer)通过在公开信道上广播一条消息来共享一个新秘密。之后 Blundo 等人[10]将 Blakley 等人的结果推广到一般的动态存取结构中，并在文献[11]中研究了共享多个秘密的协议。Cachin[12]介绍了一般存取结构上、在计算意义上安全的秘密共享协议。该协议所生成的秘密分享大小与秘密相同，同时可以共享多个秘密，允许在线动态地添加参与者，而无须向原有参与者重新分发新的秘密分享。这些功能是通过在公共准入区域(如布告栏)存储可信的公共信息来实现的。协议的安全性由单向函数的安全性来保证。本节主要介绍 Cachin 的协议[12]。

为简单起见，假设秘密 s 取自于一个有限交换群 $\boldsymbol{G}=<G,+>$，$l=\log_2|G|$，$f:G\to G$为 G 上的单向函数。

一个计算意义上安全的秘密共享协议是分发者 D 和参与者集合 P 的成员之间来共享秘密 s 的一个协议，对应的存取结构为 Γ，并满足以下条件：

(1) 分发者 D 将秘密分享 y_i 秘密地传送给 P_i，$i=1,\cdots,n$。

(2) 对于所有的授权子集 $X\in\Gamma$ 来说，他们能够利用所持有的秘密分享$\{y_i|P_i\in X\}$有效地计算出秘密 s。

(3) 对于任意的非授权子集 $X\notin\Gamma$ 来说，他们运行任意的多项式时间算法都不能以不可忽略的概率计算出秘密 s。

Cachin 的协议用到了公共准入区域，并把这个区域称为布告栏，分发者 D 可以在布告栏里发布一些不可伪造的公共信息。如果通信成本不高的话，D 也可以以消息广播的形式来代替集中式存储。一般地，上述布告栏在所有的秘密共享协议都存在，它至少包含参与者人数 n 和存取结构 Γ 等信息。

共享秘密 $s\in G$ 的基本协议如下：

(1) D 按照均匀分布随机地在 G 上选取 n 个元素 $y_1,\cdots,y_n$。

(2) D 将 y_i 秘密地传送给 P_i，$i=1,\cdots,n$。

(3) 对于每一个极小授权子集 $X\in\Gamma_0$，D 计算。

$$T_X=s-f\Big(\sum_{x:P_x\in X}y_x\Big)$$

并将 $\boldsymbol{T}=\{T_X|X\in\Gamma_0\}$发布在布告栏上。

上面涉及的加法和减法都是在 $\boldsymbol{G}$ 中运算的。授权子集 Y 恢复秘密 s 的步骤如下。

(1) Y 中的所有成员共同商议决定选出一个极小授权子集 $X \subseteq Y$。

(2) X 中的所有成员将他们的秘密分享加在一起，得到 $V_X = \sum_{x:P_x \in X} y_x$，计算 $f(V_X)$。

(3) X 从布告栏中获取 T_X，计算出秘密 $s = T_X + f(V_X)$。

很容易验证上述基本协议的完备性，即任何授权子集 $X \in \Gamma$ 都能够有效地恢复秘密 s。

下面简单分析一下协议的安全性。秘密 s 和秘密分享之间的关系可以由下面 $|\Gamma_0|$ 个方程给出：对于所有的 $X \in \Gamma$，$V_X = \sum_{x:P_x \in X} y_x$ 有

$$s = T_X + f(V_X)$$

后面指定 $V_X = \sum_{x:P_x \in X} y_x$，其中 X 为任意的参与者集合。任意的非授权子集 $U \notin \Gamma$ 都不能直接计算得到 $V_X, X \in \Gamma_0$。因此 U 仅仅利用一个方程无法计算出 s。U 也可以通过 s 或 V_X 联立两个或多个方程。

通过 s，联立两个方程得到以下关系：

$$T_Y - T_Z = f(V_Z) - f(V_Y) \quad Y, Z \in \Gamma_0$$

而对于等式右边来说，U 是无法直接计算的，除非 $V_Y = V_Z$，而此时 $T_Y = T_Z$，对于 U 来说毫无用处。

通过 V_W 联立两个方程，得到下面的关系，即

$$f^{-1}(s - T_{W \cup U'}) - f^{-1}(s - T_{W \cup U''}) = V_{U'} - V_{U''}$$

其中，$W \cap U = \varnothing$ 且 $W \cup U' \in \Gamma_0$，$W \cup U'' \in \Gamma_0$，$U' \subset U$，$U'' \subset U$ 且 $U' \neq U''$。而这对他们同样毫无用处。因此 U 无法有效地计算出秘密 s。

一般地，T 和布告栏的大小为 $O(2^n)$。如果布告栏的大小有限制，T 的可信性仍可通过分发者 D 对 T 进行数字签名来保证。如果 T 很大，则只需对 T 的一部分进行签名即可，避免了为了认证一个元素而对整个 T 进行查询。

由于任何一个授权子集在恢复秘密时，只需要获取 T 中对应的一个元素即可，因此布告栏可以看做是一个服务器，收到查询请求后返回相应的元素及对应的签名。

上述基本协议也允许分发者在分发完秘密分享之后更改所共享的秘密，而仅需更改布告栏上的 T 即可。

在很多情况下，在一个秘密的有效期内，秘密共享协议的参与者并不是一成不变的，而存取结构本身也有可能发生改变，如有新的参与者加入。类比存取结构的单调性，假设存取结构的改变也是单调的，即只有参与者加入且授权子集仍为授权子集。

一个计算意义上安全的在线秘密共享协议是在分发者 D 和参与者集合序列 $P^{(0)}$，$P^{(1)}$，…中的成员之间共享秘密 s 的一个协议，对应的存取结构序列为 $\Gamma^{(0)}$，$\Gamma^{(1)}$，…，其中对于所有的 $t \geqslant 0$ 有，$P^{(t)} \subset P^{(t+1)}$，$\Gamma^{(t)} \subseteq \Gamma^{(t+1)}$。该协议满足：

(1) $P_i \in P^{(0)}$ 对应的秘密分享 y_i 形成一个共享秘密 s、对应存取结构 $\Gamma^{(0)}$，并且在计算意义上安全的秘密共享协议。

(2) 在时间 $t > 0$，分发者 D 将秘密分享 y_i 秘密地分发给每个参与者 $P_i \in P^{(t)} \setminus P^{(t-1)}$。

(3) 对于所有的 $t \geqslant 0$，每一个授权子集 $X \in \Gamma^{(t)}$ 都可以利用 $\{y_i | P_i \in X\}$ 有效地计算出秘密 s。

(4) 对于所有的 $t \geqslant 0$，所有的非授权子集 $X \notin \Gamma^{(t)}$，运行任意的多项式时间算法都不能

以不可忽略的概率计算出秘密 s。

前面介绍的基本协议就满足上述定义，如果分发者一步一步地操作，在时间$t>0$，D 随机选取 y_i，然后秘密地发给每个新加入的参与者 $P_i \in P^{(t)} \setminus P^{(t-1)}$，并相应地将 T_X 发布在布告栏上，其中 $X \in \Gamma_0^{(t)}$ 且 $X \notin \Gamma_0^{(t-1)}$。$D$ 之前分发的秘密分享仍然有效，且不需要重传。

基本协议的简便灵活，使得它可以扩展到其他一些情况中，如在一个在线秘密共享协议中有参与者退出的情况等。与传统的秘密共享协议相比，Cachin[12] 提出的方案更加简单灵活，秘密分享大小比较小。区别就在于 Cachin[12] 方案达到的是计算意义上的安全，其安全性依赖于单向函数的安全性，并且他们在安全模型中使用了布告栏来提供公共信息。当然为了防止穷举搜索攻击，要求秘密的取值空间不能太小。

在线秘密共享协议在参与者以及存取结构或者秘密本身经常改变的情况下有很多实际的应用。当有新的参与者加入或者有参与者离开时，并不需要秘密地分发新的秘密分享。这种情况经常出现在密钥管理、密钥托管及公平公钥密码体制中。

7.7 可验证秘密共享协议

秘密共享协议解决了在遭遇被动攻击的情况下秘密的机密性和可用性之间的矛盾，但没有考虑以下两种主动攻击的情况。

(1) 欺骗的参与方：在多个参与方协同合作恢复秘密的过程中，如果有参与方提交的秘密分享是伪造的，那么秘密恢复过程的最后将得不到正确的结果。

(2) 欺骗的分发者：参与方无法判断由保密信道得到的来自秘密分发者的秘密分享是否是正确的秘密分享。

可验证秘密共享协议的概念是由 Chor 等人[13] 于 1985 年首次提出的，用以解决上述问题。

可验证秘密共享(Verifiable Secret Sharing，VSS)协议由秘密分享生成(记为 Share)算法，秘密恢复(记为 Recover)算法和一个用于检验秘密分享正确性的验证算法(记为 Verify)组成，其中

(1) 秘密分发者运行 Share 算法 Share$(S)=(y_1,y_2,\cdots,y_n)$，得到秘密 S 的秘密分享，参与方 P_i 通过安全信道得到 $y_i(i=1,2,\cdots,n)$。

(2) 参与方执行 Recover 算法求解秘密 S。

(3) Verify 算法用于检验秘密分享的正确性。

上述定义的协议满足：

(1) $\exists u \forall A \in \Gamma$:$(\forall i \in A$:Verify$(y_i)=1) \Rightarrow$ Recover$(\{y_i \mid i \in A\})=u$，并且若秘密分发者是诚实的，则 $\boldsymbol{u}=S$。即如果参与恢复秘密 S 的所有秘密分享都是正确的，那么将得到唯一的结果 $\boldsymbol{u}$(在诚实分发者的条件下，$\boldsymbol{u}=S$)。

(2) $\forall A \notin \Gamma$：Recover$(\{y_i \mid i \in A\})$无法恢复秘密 S。

如果 Verify 算法不要求参与方之间的交互，称其为**非交互的可验证秘密共享协议**。

7.7.1 Feldman 可验证秘密共享协议

Feldman 可验证秘密共享协议[14] 是 Shamir 秘密共享协议的一种推广，但可以抵抗恶

意敌手攻击，这里恶意敌手的能力被界定为：敌手可以是包括可信中心在内的任何一方，而且可以收买或控制至多$(n-1)/2$个分享者。

Feldman 可验证秘密共享协议中使用了参数p、q、g，满足$g^q \equiv 1 (\bmod\ p)$，p和q是两个大素数。

可信中心首先随机产生一个Z_q上的t次多项式$S(x)$，且满足$s=S(0)$，s是共享秘密；然后每一个分享者$i(i=1,2,\cdots,n)$得到来自可信中心的分享$s_i=S(i) \bmod q$；可信中心同时要广播数值$Vs_k=g^{a_k} \bmod p$，这里a_k是$S(x)$的$k \leqslant t$次项系数。这样每个分享者i都可以通过检验以下等式：

$$g^{s_i} \equiv \prod_k (Vs_k)^{i^k} (\bmod\ p) \left(\equiv \prod_k g^{a_k i^k} (\bmod\ p)\right)$$

是否成立来检查获得分享的可信中心的真实性。如果以上等式不满足，用户i可以要求可信中心揭露该分享(称为一个投诉，Complaint)。如果超过t个用户发出投诉，那么可信中心被宣称为不合格(Disqualified)。

在共享秘密重构时，Feldman 可验证秘密共享协议可以检测出不正确的分享s_i'。但必须指出，Feldman 可验证秘密共享协议只具有计算安全性，因为对应秘密s的值$g^{a_0}=g^s$被泄露。

7.7.2 Pedersen 可验证秘密共享协议

Pedersen 可验证秘密共享协议[15]使用参数p、q、g，满足$g^q \equiv 1 (\bmod\ p)$，p和q是两个大素数。此外，还引入参数$h \in Z_p^*$，且离散对数$\log_g(h)$是难处理的。

可信中心首先选择2个t次多项式$S(x)$、$S'(x)$，系数均在Z_q上均匀分布。共享秘密$s=S(0)$。可信中心给每个分享者$i(=1,2,\cdots,n)$分发2个分享：$s_i=S(i) \bmod q$，$s'(i)=S'(i) \bmod q$。对于两个多项式的系数做出以下承诺：

$$Vs_k = g^{a_k} h^{b_k}$$

这里a_k、b_k分别是多项式$S(x)$、$S'(x)$的$k(\leqslant t)$次项系数。

每个分享者i可以通过检验以下等式

$$g^{s_i} h^{s_i'} \equiv \prod_k (Vs_k)^{i^k} (\bmod\ p)$$

是否成立来验证自己分享的真伪。以下部分同 Feldman 可验证秘密共享协议类似。

在共享秘密重构阶段，要求所有分享者同时提交2个分享：$s_i=S(i) \bmod q$，$s'(i)=S'(i) \bmod q$，这时上述等式可以用来检验分享者所提交的分享的合法性。注意：除非恶意的分享者能够计算离散对数$\log_g(h)$，否则无法伪造虚假的分享。

Pedersen 可验证秘密共享协议满足无条件安全性，因为所泄露的信息只是$g^{a_0} h^{b_0} = g^s h^{b_0}$。可以看出，对于每一个$s$值，恰有一个$b_0$使得等式

$$Vs_0 = g^s h^{b_0}$$

成立，因此公开Vs_0并未泄露秘密s的任何信息。

7.7.3 公开可验证秘密共享协议

上述介绍的可验证秘密共享协议中，参与方只能验证自己的秘密分享的正确性，但无法得知其他参与方秘密分享的正确性。公开可验证秘密共享(Public Verifiable Secret

Sharing)协议为参与方检测秘密恢复过程中可能的欺骗提供了一种方法，从而解决了这一问题。

选取大素数 p，满足 $q=(p-1)/2$ 亦为素数。选取 p 阶循环群 G 及其生成元 $g\in G$。

一般存取结构上的可验证秘密共享协议：

令 $D=g^s$ 公开。秘密分发者为了将秘密 $s\in Z_p$ 按照存取结构 Γ 共享，对任意的 $A=\{P_{j_1},P_{j_2},\cdots,P_{j_k}\}\in\Gamma$，执行：

(1) 计算秘密分享 $y_{A_i}=\begin{cases}\text{在 } Z_p \text{ 上随机选取}, i=j_1,j_2,\cdots,j_{k-1}\\ s-\sum_{l=1}^{k-1}y_{A_{j_l}} \bmod p, i=j_k\end{cases}$。

(2) 将 y_{A_i} 通过安全信道传送给参与方 P_i，$i=j_1,j_2,\cdots,j_{k-1}$。

(3) 公开 $D_{A_i}=g^{y_{A_i}}\in G$，则所有参与方都可以验证 $\forall A\in\Gamma$：$\prod_{i\in A}D_{A_i}=g^s=D$。

参与方 P_i 通过检验 $D_{A_i}=g^{y_{A_i}}$ 来判断秘密分享 y_{A_i} 是否正确。

由于上述协议的计算量较大，对于门限存取结构的情形，可以采用前面介绍的 Feldman 协议或 Pedersen 协议。

验证协议：

为了使秘密分享可公开验证，这里采用 ElGamal 公钥加密算法对秘密分享进行加密。

选取 $h\in Z_p^*$ 为 q 阶元素。对于 $i=1,2,\cdots,n$，参与方 P_i 随机选取加密密钥 $z\in Z_q$，公开 $e=h^z \bmod p$。为了加密消息 $M\in Z_p^*$，秘密分发者随机选取 $\alpha\in Z_q$ 计算 $(h^\alpha, M^{-1}\cdot e^\alpha) \bmod p$。参与方 P_i 根据接收到的密文 (A,B)，可通过计算 $M=A^z/B \bmod p$ 进行解密。

验证协议通过验证协议中的 (A,B) 是关于 $V=g^v\in G$ 的离散对数 v 的密文来证明秘密分享的正确性。该协议基于以下事实：

$$(A,B)=(h^\alpha, v^{-1}\cdot e^\alpha) \bmod p \Rightarrow V^B=g^{vB}=g^{(e^\alpha)}$$

证明者要向验证者证明 A 关于 h 的离散对数等于 V^B 关于 g 和 e 的双重离散对数。

证明者		验证者
重复 K 次		
$w\in_R Z_q$		
$t_h=h^w \bmod p$		
$t_g=g^{(e^w)}$		
	$\xrightarrow{t_h,t_g}$	
	$\xleftarrow{c}$	$c\in_R\{0,1\}$
$r=w-c\cdot\alpha \bmod q$	$\xrightarrow{r}$	
		$t_h\overset{?}{=}h^rA^c \bmod p$
		$t_g\overset{?}{=}\begin{cases}g^{(e^r)}, & c=0\\ V^{(B\cdot e^r)}, & c=1\end{cases}$

利用 Hash 函数，可以根据上述协议构造一个非交互的验证算法：假设存在强 Hash 函数 $H_l:\{0,1\}^*\to\{0,1\}^k(k\approx100)$，对于 $i=1,2,\cdots,k$ 证明者选择 $w_i\in_R Z_q$ 计算 $t_{h,i}=h^{w_i} \bmod p$ 和 $t_{g,i}=g^{(e^{w_i})}$，然后计算 $\boldsymbol{k}$ 向量

$$R=(r_1,r_2,\cdots,r_l)=((w_1-c_1\alpha \bmod q),(w_2-c_2\alpha \bmod q),\cdots,\\(w_l-c_l\alpha \bmod q))$$

其中 c_i 代表 $c=H_k(V\parallel A\parallel B\parallel t_{h,1}\parallel t_{g,1}\parallel\cdots\parallel t_{h,k}\parallel t_{g,k}||)$ 的第 i 位。相应的验证方需要检验 $t_{h,i}=h^{r_i}A^{c_i} \bmod p$ 和 $t_{g,i}=(g^{(1-c_i)}V^{(c_iB)})^{(e^{r_i})}, i=1,2,\cdots,k$。

这里介绍了 Stadler 于 1996 年提出的基于离散对数计算上的复杂性的公开可验证秘密共享协议[16]。Stadler 同时提出的另一个基于大数分解问题的困难性的公开可验证秘密共享协议与上述协议类似,此处不再赘述。感兴趣的读者可参阅文献[16]。

7.8 无可信中心的秘密共享协议

很多应用场合都不存在可信中心,如密钥分配协议中不需要密钥分配中心 KDC 的存在,这时,需要对前面介绍的秘密共享协议进行重新设计和修改。

无可信中心的秘密共享协议的基本思想是:每个分享者 i 都运行相同的一个 Shamir 秘密共享协议或 Pedersen 可验证秘密共享协议,对应的秘密也由各分享者自行选取。实际上,这时每个用户都是一个可信中心。最终共享的秘密就是这些子秘密之和。

下面以 Shamir 秘密共享协议为例进行说明。

具体过程如下:

(1) 每个分享者 $P_i(i=1,2,\cdots,n)$ 随机选取秘密 s^i,随机选取 t 次多项式 $h_i(x)$(常数项就是 s^i)。

(2) 每个分享者 $P_i(i=1,2,\cdots,n)$ 分别把 $y_j^i=h_i(j)$ 分发给其他分享者 P_j。

(3) 每个分享者 P_j 的分享就是 $\sum_{i=1}^{n}y_j^i$。

显然,最终的共享秘密 $s=\sum_{i=1}^{n}s_i$,在等价意义上,用于重构的多项式就是 $\sum_{i=1}^{n}h_i(x)$。秘密重构公式同前,这里不再赘述。

类似地,可以自然地推广到 Pedersen 可验证秘密共享协议的情况。

7.9 前摄秘密共享协议

尽管上述介绍的各种秘密共享协议的安全性在理论上是足够的,但在现实生活中,随着时间的流逝,系统中各个部件所受到的攻击也不断增加,部件所持有的秘密分享信息有可能因为各种原因被破坏、受损伤甚至泄露。尽管人们可以建立非常强的安全保护机制,但在时间面前,一个极小的缺陷也会被慢慢放大,发生错误的可能性也会逐渐增加。所以,人们提出了前摄(Pro-Active)机制来解决这个问题。

所谓"前摄"是指在保护的秘密 s 不变的情况下,被分发给每个参与者的部分秘密分享信息可以被随时更新。这样,即使单个的部分秘密分享信息被泄露,在更新之后,泄露的信息就变得毫无用处。只要周期性地更新秘密分享信息,就可以在不改变共享秘密的情况下,大大提高系统的安全性。

前摄的另外一个用处是修复损坏的秘密分享信息,假设 n 个参与者中的 t 个联合才可

以恢复秘密，那么随着时间的流逝，一旦被损坏的秘密分享信息逐渐增多，使得最终持有正确秘密分享信息的参与者小于 t 个，就再也无法恢复秘密了。因此，周期性地更新可以修复损坏的秘密分享信息，有效地防止这种情况的出现。

本节以 Shamir 秘密共享协议为例，说明基本的前摄秘密共享思想。在这个协议中不存在分发中心，目的是在没有分发中心参与的情况下更新秘密分享信息，所有的分享者应自行协商一个新的多项式，这个多项式保存的秘密和旧的多项式一样都是 s。在协议执行后，每个分享者将得到一个新的秘密分享信息，这个秘密分享信息属于新的 $t-1$ 次多项式。

假设使用 Shamir 秘密共享协议，秘密 s 被保存在一个 $t-1$ 次多项式 $f(x)$ 中，分发给 n 个分享者 $P_1, P_2, \cdots, P_n$，每个分享者持有秘密分享信息 $f(i)$。

具体的秘密分享信息更新协议如下：

(1) 每个分享者 P_i 从 Z_p 中随机选择 $t-1$ 个数，形成一个 $t-1$ 次多项式 $p_i(x)$，其常数项为 0，即 $p_i(0)=0$。

(2) 每个分享者 P_i 对 P_j 分发秘密分享信息 $p_i(j)$。

(3) 每个分享者 P_i 接收到以下的秘密分享信息 $p_1(i), p_2(i), \cdots, p_n(i)$，并把旧的秘密分享信息 $f(i)$ 和新的 n 个秘密分享信息相加得到新的秘密分享信息 $h(i)$，即

$$h(i) = f(i) + \sum_{c=1}^{n} p_c(i)$$

(4) 每个分享者 P_i 销毁其旧的秘密分享信息 $f(i)$。

从这个协议可以看出，n 个分享者实际上重新构造了一个新的 $t-1$ 次多项式 $h(x)$ 为

$$h(x) = f(x) + \sum_{c=1}^{n} p_c(x)$$

并且 $h(0) = f(0) + \sum_{c=1}^{n} p_c(0) = s + 0 = s$，所以这个新的多项式 $h(x)$ 仍然满足 Shamir 秘密共享协议的要求，每个分享者手中也拥有一个 $h(x)$ 上的点。所以，新的秘密分享信息完全符合 Shamir 秘密共享协议。

7.10 小结

秘密共享的基本思想是由 Shamir[1] 和 Blakley[17] 分别独立提出来的。秘密共享协议及其基本思想在信息安全中发挥着重要作用，是设计其他安全协议的一个基础性工具和技术，本章简要介绍了一些基本的、典型的秘密共享协议。这里需要提醒的是 7.4 节和 7.5 节对于某些读者来说阅读起来可能比较困难，建议做一般了解即可。关于秘密共享协议的构造方法、结构和熵界等方面的研究工作很多，感兴趣的读者可参阅文献[18]～[25]。

本章在写作过程中得到了周展飞副教授的大力支持，他提供了大量的相关材料，作者在此表示衷心的感谢。

参 考 文 献

[1] Shamir A. How to share a secret, Comm. ACM, Vol.22(11), 1979, 612～613.

[2] Asmuth C, Bloom J. A Modular approach to key safeguarding, IEEE Transactions on Information

Theory, Vol. IT-30, Mar 1983,208～210.

[3] Ito M, Saito A, Nishizeki T. Secret sharing scheme realizing general access structure, in Proceedings of IEEE Globecom'87,99～102, 1987.

[4] Cramer R, Fehr S. Optimal black-box secret sharing over arbitrary Abelian groups. M. Yung, editor, Advances in Cryptography—Crypto'02, volume 2442 of Lecture Notes in Computer Science, 272～287. Springer-Verlag, 2002.

[5] Cramer R, Fehr S, Stam M. Black box secret sharing from primitive sets in algebraic number fields, CRYPTO 2005. LNCS 3621,344～360, 2005.

[6] Chor B, Kushilevitz E. Secret sharing over infinite domains, J. Cryptology 6 (1993), 87～95 [Preliminary version appeared in "Advances in Cryptology—CRYPTO'89", G. Brassard, ed., Lecture Notes in Computer Science 435 (1990),299～306].

[7] Benaloh J, Leichter J. Generalized secret sharing and monotone functions, Advances in Cryptography—Crypto 86 (Proceedings), A. M. Odlyzko (ed.), 213～222, Lecture Notes in Computer Science, Vol. 263, Springer-Verlag, Berlin, 1987.

[8] Blakley G R, Swanson L. Security proofs for information protection systems, Proc. IEEE Symp. on Security and Privacy, 75～88, 1981.

[9] Blakley B, Blakley G R, Chan A H. Threshold schemes with disenrollment, Advances in Cryptology—CRYPTO '92, E. F. Brickell, ed., vol. 740 of Lecture Notes in Computer Science, 540～548, Springer-Verlag, 1993.

[10] Blundo C, Cresti A, DE Santis A. Fully dynamic secret sharing schemes, Advances in Cryptology—CRYPTO '93, D. R. Stinson, ed., vol. 773 of Lecture Notes in Computer Science, 110～125, Springer-Verlag, 1994.

[11] Blundo C, DE Santis A, DI Crescenzo G. Multi-secret sharing schemes, Advances in Cryptology—CRYPTO '94, Y. G. Desmedt, ed., vol. 839 of Lecture Notes in Computer Science, 150～163, Springer-Verlag, 1994.

[12] Cachin C. On-line secret sharing, Proceedings of 5th IMA Conference on Cryptography and Coding, 190～198, Springer-Verlag, Berlin, 1995.

[13] Chor B, Goldwasser S, Micali S, Awerbuch B. Verifiable secret sharing and achieving simultaneity in the presence of faults. In Proceedings of 26th IEEE Symposium on the Foundations of Computer Science (FOCS), 383～395, 1985.

[14] Feldman P. A Practical Scheme for Non-Interactive Verifiable Secret Sharing. In: Proc. 28th IEEE Symposium on Foundations of Computer Science(FOCS'87), Toronto, Canda, 1987. 427～437.

[15] Pedersen T. Non-Interactive and information-theoretic Secure Verifiable secret sharing. In: Feigenbum(Ed.). Advances in Cryptology-Crypto'91, LNCS No. 576. Berlin: Springer-Verlag Heidelberg, 1992. 129～140.

[16] Stadler M. Public Verifiable Secret Sharing. In Advances in Cryptology-EUROCRYPT'96, volume 1070 of Lecture Notes in Computer Science, 190～199. Springer-Verlag, 1996.

[17] Blakley G R. Safeguarding cryptographic keys. Federal Information Processing Standard Conference Proceedings, 48 (1979), 313～317.

[18] Blundo C, De Santis A, Stinson D R. Graph decompositions and secret sharing schemes. Lecture Notes in Computer Science, 658 (1993), 1～24. (Eurocrypt'92.)

[19] Brickell E F. Some ideal secret sharing schemes. Journal of Combinatorial Mathematics and Combinatorial Computing, 9 (1989),105～113.

[20] Brickell E F, Davenport D M. On the classification of ideal secret sharing schemes. Journal of Cryptology, 4 (1991), 123～134.

[21] Capocelli R M, De Santis A, Gargano L. On the size of shares for secret sharing schemes. Journal of Cryptology, 6(1993),157～167.

[22] Simmgns G J. An introduction to shared secret and/or shared control schemes and their application. In Contemporary Cryptology, The Science of Information Integrity,441～497. IEEE Press, 1992.

[23] Stinson D R. An explication of secret sharing schemes. Designs, Codes and Cryptography, 2 (1992), 357～390.

[24] Stinson D R. Decomposition constructions for secret sharing schemes. IEEE Transactions on Information Theory, 40 (1994), 118～125.

[25] 刘木兰,张志芳.密钥共享体制和安全多方计算.北京:电子工业出版社,2008.

第 8 章　数字签名协议

政治、军事、外交等活动中签署文件，商业上签订契约和合同以及日常生活中在书信、从银行取款等事务中的签字，传统上都采用手写签名或印鉴。签名起到认证、核准和生效作用。随着社会信息化程度的提高，人们期望通过数字通信网对贸易合同等进行迅速的、远距离的签名，数字签名或电子签名应运而生，它是以电子形式存储的一种消息，可以在通信网络中传输。数字签名已开始用于商业通信系统，诸如电子邮件、电子转账、办公室自动化等系统中。从社会应用角度来看，数字签名的应用也相对比较成熟，很多国家和地区都已经出台了相应的法律来支持数字签名或电子签名。我国也于 2004 年正式出台了电子签名法。

手写签名与数字签名的主要差别在于以下几点。

(1) 签署文件方面的不同。手写签名是所签文件的物理部分，数字签名则不是，所以数字签名必须设法把签名“绑”到所签文件上。

(2) 验证方面的不同。手写签名是通过和作为标准的真实的手写签名相比较来验证，而数字签名则是通过一个公开的验证算法来验证。安全的数字签名将阻止伪造签名的可能性，相比之下，手写签名比较容易伪造。

(3) “复制”方面的不同。手写签名不易复制，因为不难将文件的副本与原文件区别开来。而一个数字签名很容易复制，电子文件的副本与原文件没有任何区别。这个特点要求必须防止一个数字签名消息的重复使用，一般通过要求消息本身包含诸如日期等信息来达到这一目的。

一个数字签名协议至少应满足以下 3 个条件：①签名者事后不能否认自己的签名；②任何其他人都不能伪造签名，接收者能验证签名；③当双方签名的真伪发生争执时，法官或第三方能解决双方之间发生的争执。

基于公钥密码算法和对称密码算法都可以设计数字签名协议。特别是公钥密码算法的诞生为数字签名的研究与应用开辟了一条广阔的道路。目前诸多数字签名协议主要基于公钥密码算法。

数字签名协议主要有两类：一类是普通数字签名协议，只提供签名生成和签名验证两个基本功能，而且这两个功能可分别由单个签名者和单个验证者独立完成，该类数字签名协议通常称为数字签名算法，如第 1 章中介绍的 RSA 数字签名算法、DSA；另一类是特殊数字签名协议，该类数字签名协议除了提供签名生成和签名验证两个基本功能外，还具有其他特定的辅助功能或需要联合签名或验证，如不可否认的数字签名协议、Fail-Stop 数字签名协议、群数字签名协议、盲数字签名协议等。本章主要介绍第二类有代表性的数字签名协议。

通常，一个普通数字签名协议即数字签名算法主要由两个算法组成，即签名算法和验证算法。签名者使用一个(秘密)签名算法 Sig(·)得到对消息 x 的签名 $y=\mathrm{Sig}(x)$。x 的签名 y 是否真实，可通过一个公开的验证算法 Ver(·,·)来回答。一个普通数字签名协议可由满足下列条件的五重组(P,A,K,S,V)来描述。

(1) P 是由所有可能的消息组成的一个有限集合。

(2) A 是由所有可能的签名组成的一个有限集合。

(3) K 是由所有可能的密钥组成的一个有限集合，即密钥空间。

(4) 对每一个 $k \in K$，有一个秘密的签名算法 $\mathrm{Sig}_k(\cdot) \in S$ 和一个对应的公开的验证算法 $\mathrm{Ver}_k(\cdot,\cdot) \in V$。每一个 $\mathrm{Sig}_k(\cdot): P \to A$ 和 $\mathrm{Ver}_k(\cdot,\cdot): P \times A \to \{$真，假$\}$是满足下列条件的函数：对每一个消息 $x \in P$ 和每一个签名 $y \in A$，有 $\mathrm{Ver}_k(x,y)=$真，当且仅当 $y=\mathrm{Sig}_k(x)$，$\mathrm{Sig}_k(\cdot)$和 $\mathrm{Ver}_k(\cdot,\cdot)$都是多项式时间函数。

利用上述描述方法可重新描述第 1 章中介绍的数字签名算法。这里就不再赘述，留给读者自行完成。

8.1 潜信道签名协议

一个值得注意的问题是在数字签名协议中，很容易构建潜信道。潜信道（Subliminal Channel）这一概念[1]是由 Simmons 于 1983 年首次提出的，它的基本思想是通信双方通过交换完全无关的、签了名的消息来传送秘密消息，使得第三方即使看到他们的所有通信也无法知道寓于通信中的秘密信息。一个简单的潜信道可以是句子中单词的个数，句子中有奇数个单词对应“1”，有偶数个单词对应“0”。因此，当读这种仿佛无关的段落时，已将 0、1 串消息发送给了在场的我方的人。当然，这种算法不安全。设计安全性好的潜信道协议是可能的，现在已有一些具体协议。潜信道签名协议与通常的签名协议没有什么不同，在潜信道协议中，第三方不仅不能读懂潜消息，而且他也不知道存不存在潜信道。潜信道的一个明显的应用是在间谍网中。如果每人都收发签名的消息，间谍在签名消息时发送潜消息就不会被注意到。

这里介绍一个基于 ElGamal 签名算法的潜信道签名协议[2]。

密钥的产生类似于基本的 ElGamal 签名协议，即 ElGamal 签名算法。首先选择一个素数 p、两个随机数 g 和 r，使得 g 和 r 都小于 p。然后计算 $K=g^r \bmod p$。公开 K、g 和 p，保密 r。该通信双方分别为 A 和 B，他们都知道 r，这个密钥一方面用来发送和阅读潜消息，另一方面还用来对消息签名。

为使用消息 M' 来发送潜消息 M，要求 $\gcd(M,p)=1$，$\gcd(M',p)=1$，$\gcd(M,p-1)=1$。发送方 A 计算 $x=g^M \bmod p$，并由方程 $M' \equiv (rx+My)(\bmod (p-1))$求出 y，像在基本的 ElGamal 签名协议中一样，M'的签名是(x,y)。所有的人都能通过验证方程 $K^x x^y \equiv g^{M'} (\bmod p)$来验证签名的合法性。如果 B 收到的(x,y)是 M'的一个合法签名，那么他可以恢复潜消息 M，恢复公式为

$$M = y^{-1}(M' - rx) \bmod (p-1)$$

由上述讨论可知，基本的 ElGamal 数字签名协议可嵌入潜信道。人们自然会想到 DSA 是否可嵌入潜信道。Simmons 于 1993 年发现了 DSA 中的一种潜信道[3]。他认为，DSA 提供了至今发现的最适宜于潜通信的环境。NIST 和 NSA 还未对此潜信道作出评价，没有人知道他们是否早就知道潜信道。因为潜信道使不诚实的 DSA 实现者可以在每次签名时泄露密钥的一部分，从而对签名方案构成威胁。使用者在选用别人给你实现的 DSA 时要十分慎重。关于 DSA 的潜信道的详细讨论参见文献[3]。

文献[1]和[4]中还介绍了在签名协议中嵌入潜信道的其他协议。文献[5]中还讨论了

在下一章将要介绍的 Fiat-Shamir 和 Feige-Fiat-Shamir 识别协议中嵌入潜信道的协议以及对其可能的滥用。事实上，任意签名协议都能转化成潜信道协议。

8.2 不可否认的数字签名协议

不可否认的数字签名(Undeniable Digital Signature)是由 Chaum 和 Antwerpen 在1989 年提出的[6]。不可否认的数字签名是一种特殊的数字签名，它具有一些新颖的特征，没有签名者的合作，接收者就无法验证签名，在某种程度上这种签名保护了签名者的利益。这种签名在某些应用场合是十分有用的。例如，软件开发者可利用不可否认的数字签名对他们的软件进行保护，使得只有付了钱的顾客才能验证签名并相信开发者仍然对软件负责。

一个不可否认的数字签名的真伪性是通过接收者和签名者执行一个协议来推断的(假定签名者参加了这个协议)，这个协议称为否认协议(Disavawal Protocol)。因为签名者可声称一个合法的签名是伪造的，在这种情况下，如果签名者拒绝参加验证，就可认为签名者有欺骗行为。如果签名者参加验证，由否认协议就可推断出签名者的真伪性。自从不可否认的数字签名诞生以来，人们围绕这种签名讨论了一系列相关的签名，如可转移的不可否认的数字签名、零知识的不可否认的数字签名等。

一个不可否认的数字签名协议由以下 3 部分组成：一个签名算法、一个验证协议和一个否认协议。现在首先来介绍 Chaum-Van Antwerpen 不可否认的数字签名协议的签名算法和验证协议。

设 $p=2q+1$ 是一个使得 q 是素数并且在 Z_p 上的离散对数是难处理的素数，$\alpha\in Z_p^*$ 是一个阶为 q 的元素，$1\leqslant a\leqslant q-1$，令 $\beta=\alpha^a \bmod p$。设 G 表示 Z_p^* 中的阶为 q 的乘法子群(G 由模 p 的二次剩余构成)，设 $P=A=G$，定义 $K=\{(p,\alpha,a,\beta)\mid\beta\equiv\alpha^a \pmod p\}$。值 p、α 和 β 是公开的，a 是秘密的。

对 $K=(p,\alpha,a,\beta)$ 和 $x\in G$，定义 $y=\mathrm{Sig}_K(x)=x^a \bmod p$。对 $x,y\in G$，通过执行下列协议来验证签名：

(1) 接收者 A 随机选择 $e_1,e_2\in Z_q^*$。

(2) A 计算 $c=y^{e_1}\beta^{e_2} \bmod p$，并将 c 发送给签名者 B。

(3) B 计算 $d=c^{a^{-1} \bmod q} \bmod p$，并将 d 发送给 A。

(4) A 将 y 作为合法的签名接收，当且仅当 $d\equiv x^{e_1}\alpha^{e_2} \pmod p$。

下面就来证明 B 只能以很小的概率欺骗 A。这个结果不依赖于任何计算假设，也就是说，安全性是无条件的。

定理 8.1 如果 $y\neq x^a \bmod p$，那么接收者 A 将以 $\dfrac{1}{q}$ 的概率把 y 作为 x 的一个合法签名接收。

证明 因为 y 和 β 都是阶为素数 q 的乘法群 G 中的元素，所以每一个可能的口令 c 恰好对应于 q 个有序对 (e_1,e_2)。当签名者 B 接收到口令 c 时，他没有办法知道 c 是 A 用这 q 个可能的有序对 (e_1,e_2) 中的哪一个来构造的。下面就来说明，如果 $y\neq x^a \bmod p$，那么 B 所作的任何可能的回答 $d\in G$ 恰好和这 q 个有序对 (e_1,e_2) 中的一个一致。

因为 α 是 G 的生成元，所以能将 G 中的每一个元素表示成 α 的幂次方。设 $c=\alpha^i$，$d=$

α^j，$x=\alpha^k$，$y=\alpha^l$，$i,j,k,l\in Z_q$。考虑下列两个同余式：

$$c\equiv y^{e_1}\beta^{e_2}(\text{mod }p),\quad d\equiv x^{e_1}\alpha^{e_2}(\text{mod }p)$$

上面两个同余式等价于下面两个同余式：

$$\begin{aligned} i&\equiv le_1+ae_2(\text{mod }q)\\ j&\equiv ke_1+e_2(\text{mod }q)\end{aligned}\tag{8-1}$$

现在假定 $y\neq x^a \text{ mod } p$，这等价于 $l\not\equiv ak(\text{mod }q)$。而同余方程组(8-1)的系数矩阵为 $\begin{bmatrix} l & a\\ k & 1\end{bmatrix}$，该矩阵的行列式 $\begin{vmatrix} l & a\\ k & 1\end{vmatrix}=l-ak\not\equiv 0(\text{mod }q)$，所以方程组(8-1)有唯一解。也就是说，每一个 $d\in G$ 恰好是对 q 个可能的有序对(e_1,e_2)中之一的正确回答。因此，B 将恰以 $\frac{1}{q}$ 的概率给 A 一个能通过验证的回答 d。

现在来介绍 Chaum-Van Antwerpen 不可否认的数字签名协议的否认协议。该协议由两轮验证协议组成。步骤(1)～(4)和步骤(5)～(8)构成两轮不成功的验证协议。步骤(9)是一个一致性检测(Consistency Check)，这个检测能使 A 确定是否B 按协议中规定的方式形成他的回答。否认协议的详细执行过程如下：

(1) A 随机选择 $e_1,e_2\in Z_q^*$。

(2) A 计算 $c=y^{e_1}\beta^{e_2}\text{ mod }p$，并将 c 发送给 B。

(3) B 计算 $d=c^{a^{-1}\text{mod }q}\text{ mod }p$，并将 d 发送给 A。

(4) A 验证 $d\neq x^{e_1}\alpha^{e_2}\text{ mod }p$。

(5) A 随机选择 $f_1,f_2\in Z_q^*$。

(6) A 计算 $C=y^{f_1}\beta^{f_2}\text{ mod }p$，并将 C 发送给 B。

(7) B 计算 $D=C^{a^{-1}\text{mod }q}\text{ mod }p$，并将 D 发送给 A。

(8) A 验证 $D\neq x^{f_1}\alpha^{f_2}\text{ mod }p$。

(9) A 作出推断：y 是一个伪造，当且仅当$(d\alpha^{-e_2})^{f_1}\equiv(D\alpha^{-f_2})^{e_1}(\text{mod }p)$。

现在需说明否认协议可做到以下两点要求：一是 B 能使 A 相信一个不合法的签名是一个伪造；二是 B 以一个很小的概率使 A 相信一个合法的签名是一个伪造。

定理 8.2 如果 $y\neq x^a\text{ mod }p$，并且 A 和 B 都采纳了否认协议，那么$(d\alpha^{-e_2})^{f_1}\equiv(D\alpha^{-f_2})^{e_1}(\text{mod }p)$。

证明 因为 $d\equiv c^{a^{-1}}(\text{mod }p)$，$c\equiv y^{e_1}\beta^{e_2}(\text{mod }p)$，$\beta\equiv\alpha^a(\text{mod }p)$，所以有

$$\begin{aligned}(d\alpha^{-e_2})^{f_1}&\equiv((y^{e_1}\beta^{e_2})^{a^{-1}}\alpha^{-e_2})^{f_1}(\text{mod }p)\\ &\equiv y^{e_1f_1a^{-1}}\beta^{e_2a^{-1}f_1}\alpha^{-e_2f_1}(\text{mod }p)\\ &\equiv y^{e_1f_1a^{-1}}\alpha^{e_2f_1}\alpha^{-e_2f_1}(\text{mod }p)\\ &\equiv x^{e_1f_1}(\text{mod }p)\end{aligned}$$

同理可计算出，$(D\alpha^{-f_2})^{e_1}\equiv x^{e_1f_1}(\text{mod }p)$。

所以否认协议中步骤(9)的一致性检验是成功的。

定理 8.2 说明了否认协议可做到上述第一点要求。

B 也许企图否认一个合法的签名。在这种情况下，不假定 B 采纳了协议，也就是说，B 也许没有按协议的规定去构造 d 和 D。因此，在下面的定理 8.3 中，只假定 B 能产生满足否认协议中步骤(4)、(8)和(9)的值 d 和 D。

定理 8.3 假定 $y \equiv x^a \pmod p$，并且 A 采纳了否认协议。如果 $d \not\equiv x^{e_1}\alpha^{e_2} \pmod p$ 并且 $D \not\equiv x^{f_1}\alpha^{f_2} \pmod p$，那么 $(d\alpha^{-e_2})^{f_1} \not\equiv (D\alpha^{-f_2})^{e_1} \pmod p$ 的概率是 $1-\frac{1}{q}$。

证明 假定下列的同余式成立：$y \equiv x^a \pmod p$

$$d \not\equiv x^{e_1}\alpha^{e_2} \pmod p$$

$$D \not\equiv x^{f_1}\alpha^{f_2} \pmod p$$

$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \pmod p$$

将导出一个矛盾。

一致性检测(步骤(9))可表示成下列形式：$D \equiv d_0^{f_1}\alpha^{f_2} \pmod p$，其中 $d_0 = d^{\frac{1}{e_1}}\alpha^{\frac{-e_2}{e_1}} \bmod p$ 是仅仅依赖于否认协议中步骤(1)～(4)的一个值。

由定理 8.1 知，y 是 d_0 的合法签名的概率为 $1-\frac{1}{q}$。但是已假定 y 是 x 的合法签名，所以 $x^a \equiv d_0^a \pmod p$ 的概率为 $1-\frac{1}{q}$，这暗含着 $x = d_0$ 的概率为 $1-\frac{1}{q}$。

然而，事实 $d \not\equiv x^{e_1}\alpha^{e_2} \pmod p$ 意味着 $x \not\equiv d^{1/e}\alpha^{-e_2/e_1} \pmod p$。因为 $d_0 \equiv d^{1/e_1}\alpha^{-e_2/e_1} \pmod p$，所以有 $x \neq d_0$，这是一个矛盾。

故 B 欺骗 A 的概率为 $\frac{1}{q}$。

定理 8.3 表明，否认协议可做到上述的第二点要求。

8.3 Fail-Stop 数字签名协议

Fail-Stop 数字签名的不可伪造性也依赖于一个计算假设，具体地说，如果有人伪造了签名者的一个签名，那么该签名者能够证明作为协议依存基础的计算假设已被攻破，从而证明该签名属伪造。这个证明不依赖于任何密码假设，它也可能失败，但失败的概率独立于伪造者的计算能力，从而能保护具有多项式时间计算能力的签名者免遭具有无限计算能力的伪造者的攻击。再者，在第一次伪造之后，系统的所有参加者或系统操作人员都知道签名协议已被攻破，因此系统将终止工作，这是该系统为什么称为"Fail-Stop(失败-停止)"的原因。

一个 Fail-Stop 数字签名协议主要由 3 个算法构成，即签名算法、验证算法和"伪造证明"算法(这里"伪造证明"指对伪造签名的一种证明)。一个安全的 Fail-Stop 数字签名协议应具有下列特性。

(1) 如果签名者正确地签一个消息，那么接收者接收这个签名。

(2) 一个具有多项式界计算能力的伪造者不能构造签名使之通过验证。

(3) 具有无限计算能力的伪造者构造出得以通过验证的签名的概率极小，签名者对于一个伪造的签名能产生并出示一个"伪造证明"使得第三方相信该签名确属伪造(这一特性保证签名者的安全)。

(4) 一个具有多项式界的签名者不能构造他能对它产生"伪造证明"的非法签名(这一特性保证接收者的安全)。

下面来介绍一个 Fail-Stop 数字签名协议，该协议是 Van Heyst 和 Pedersen 于 1992 年提出的[7]，即一个密钥只能用来签名一次。首先描述 Van Heyst-Pedersen 的 Fail-Stop 数

字签名协议中的签名算法和验证算法。

设 $p=2q+1$，p 与 q 是素数，且在 Z_p 上的离散对数问题是难处理的，$\alpha\in Z_p^*$ 是一个阶为 q 的元素。设 $1\leqslant a_0\leqslant q-1$，定义 $\beta=\alpha^{a_0} \bmod p$。值 p、q、α、β 和 a_0 由一个可信中心来选定（假定系统中存在这样的中心）。p、q、α 和 β 是公开的，a_0 的值对任何人（包括 B）都是保密的。

设 $P=Z_q$，$A=Z_q\times Z_q$，签名者 B 的密钥 K 形如 $K=(\gamma_1,\gamma_2,a_1,a_2,b_1,b_2)$，这里 $a_1,a_2,b_1,b_2\in Z_q$，$\gamma_1=\alpha^{a_1}\beta^{a_2} \bmod p$，$\gamma_2=\alpha^{b_1}\beta^{b_2} \bmod p$。$B$ 公开 γ_1,γ_2，保密 a_1,a_2,b_1,b_2。

密钥 $K=(\gamma,\gamma_2,a_1,a_2,b_1,b_2)$ 对消息 $x\in Z_q$ 的签名定义为 $y=\mathrm{Sig}_K(x)=(y_1,y_2)\in Z_q\times Z_q$，这里 $y_1=(a_1+xb_1) \bmod q$，$y_2=(a_2+xb_2) \bmod q$。

用 $\mathrm{Ver}_K(x,y)$ 表示签名验证函数，易见

$$\mathrm{Ver}_K(x,y)=\text{真}\Leftrightarrow\gamma_1\gamma_2^x\equiv\alpha^{y_1}\beta^{y_2}\pmod p$$

下面讨论该协议的安全性。首先看一看有关密钥的一些基本事实。

密钥 $K_1=(\gamma_1,\gamma_2,a_1,a_2,b_1,b_2)$ 和 $K_2=(\gamma_1',\gamma_2',a_1',a_2',b_1',b_2')$ 等价是指 $\gamma_1=\gamma_1'$，$\gamma_2=\gamma_2'$。可以证明任何一个等价类中恰有 q^2 个密钥。关于等价密钥有以下一些简单事实。

引理 8.1 假定 K 和 K' 是等价的密钥，如果 $\mathrm{Ver}_K(x,y)=$真，那么 $\mathrm{Ver}_{K'}(x,y)=$真。

证明 可设 $K=(\gamma_1,\gamma_2,a_1,a_2,b_1,b_2)$，$K'=(\gamma_1,\gamma_2,a_1',a_2',b_1',b_2')$，记 $f_K(x,y)=(\gamma_1\gamma_2^x-\alpha^{y_1}\beta^{y_2})\pmod p$，类似地定义 $f_{K'}(x,y)$。易见 $f_K(x,y)=f_{K'}(x,y)$。$\mathrm{Ver}_K(x,y)=$真$\Leftrightarrow f_K(x,y)=0\Leftrightarrow f_{K'}(x,y)=0\Leftrightarrow\mathrm{Ver}_{K'}(x,y)=$真。

引理 8.2 给定密钥 K 和消息 $x\in Z_q$，那么恰有 q 个与 K 等价的密钥 K' 使得 $\mathrm{Sig}_{K'}(x)=\mathrm{Sig}_K(x)$。

证明 记 $K=(\gamma_1,\gamma_2,a_1,a_2,b_1,b_2)$，$\gamma_i=\alpha^{c_i} \bmod p$，$c_i\in Z_q$，$i=1,2$。又记 $\mathrm{Sig}_K(x)=(y_1,y_2)$。对任一密钥 $K'=(\gamma_1',\gamma_2',u_1,u_2,v_1,v_2)$，$K'$ 与 K 等价且 $\mathrm{Sig}_{K'}(x)=\mathrm{Sig}_K(x)$ 当且仅当 $\gamma_i'=\gamma_i$，$i=1,2$，且 (u_1,u_2,v_1,v_2) 适合下述方程：

$$\boldsymbol{B}(u_1,u_2,v_1,v_2)^{\mathrm{T}}=(c_1,c_2,y_1,y_2)^{\mathrm{T}}\pmod q,\quad \boldsymbol{B}=\begin{bmatrix}1 & a_0 & 0 & 0\\ 0 & 0 & 1 & a_0\\ 1 & 0 & x & 0\\ 0 & 1 & 0 & x\end{bmatrix}$$

容易检验 $\boldsymbol{B}$ 的秩为 3，$(u_1,u_2,v_1,v_2)=(a_1,a_2,b_1,b_2)$ 是方程的一个解，所以该方程的解空间的维数为 $4-3=1$，故恰有 q 个这种密钥。

引理 8.3 给定密钥 K，消息 x 与 x'，$x'\neq x$，如果密钥 K' 等价于 K，且 $\mathrm{Sig}_{K'}(x)=\mathrm{Sig}_K(x)$，$\mathrm{Sig}_{K'}(x')=\mathrm{Sig}_K(x')$，那么 $K'=K$。

引理 8.3 的证明类似于引理 8.2 的证明。

综上所述，有下述定理。

定理 8.4 给定 $\mathrm{Sig}_K(x)=y$ 和 $x'\neq x$，敌手 C 猜中 $\mathrm{Sig}_K(x')$ 的概率是 $\dfrac{1}{q}$。

定理 8.4 的安全性是无条件的，即不依赖于 C 的计算能力，因为 C 无法区别出 B 使用了使得 $\mathrm{Sig}_K(x)=y$ 的 q 个可能的密钥 K 中的哪一个。迄今为止，已经说明了，给定消息 x 的一个签名 y，C 不能计算 B 对不同的消息 x' 的签名 y'。但 C 也许能伪造一个合法的，即可通过验证的签名 $y''=(y_1'',y_2'')$，虽然 $y''\neq\mathrm{Sig}_K(x')$。然而，面对这样一个合法的伪造签名，

B 能由它算出 $a_0=\log_\alpha\beta$ 的概率是 $1-\frac{1}{q}$。因 a_0 只有可信中心知道，B 可用 a_0 作为该签名的伪造证明。下面说明 B 如何计算 a_0。

由假设，y'' 可通过验证，即 $\gamma_1\gamma_2^{x'}\equiv\alpha^{y''_1}\beta^{y''_2}(\bmod\ p)$。将 B 自己对 x' 的签名记作 $y'=(y'_1,y'_2)$，有 $\gamma_1\gamma_2^{x'}\equiv\alpha^{y'_1}\beta^{y'_2}(\bmod\ p)$。因此，$\alpha^{y''_1}\beta^{y''_2}\equiv\alpha^{y'_1}\beta^{y'_2}(\bmod\ p)$。因为 $\beta=\alpha^{a_0}\bmod p$，上式等价于 $y''_1+a_0y''_2\equiv y'_1+a_0y'_2(\bmod\ q)$，即 $y''_1-y'_1\equiv a_0(y'_2-y''_2)(\bmod\ q)$。因为 y'' 是一个伪造，依引理 8.2，$y'_2\equiv y''_2(\bmod\ q)$ 成立的概率仅为 $1/q$，所以以 $1-1/q$ 的概率存在 $(y'_1-y''_2)^{-1}\bmod q$，从而以同样的概率可求得

$$a_0=\log_\alpha\beta=(y''_1-y'_1)(y'_1-y''_2)^{-1}\bmod q$$

当然，是在签名者 B 不能计算离散对数 $\log_\alpha\beta$ 这一假定的条件之下，第三方接收 B 的这一伪造证明。

最后指出，Van Heyst-Pedersen 签名协议是一个一次签名协议，因为如果使用同一个密钥 K 签两个不同的消息，那么很容易计算出该密钥 K。

8.4 群数字签名协议

群数字签名允许群中的每个成员都能以群的名义匿名地签发消息。群数字签名具有下列 3 个特性。

(1) 只有群的成员才能代表该群签发消息。

(2) 签名的接收者能验证它是该群的合法签名，但不能揭示它是群中哪个成员所签。

(3) 在发生争端的情况下，借助于一个可信机构能识别出执行签名的成员。

一个群数字签名协议主要由 3 个算法组成，即签名算法、验证算法和识别算法(用来识别签名者)。

这种签名的一个实用的例子是投标，这里群是指由所有的提交投标的公司组成的集合，每个公司是群中成员。每个公司匿名地使用群数字签名签他的投标，当特定的投标被选中后，那个签名者能被识别出，而所有其他投标公司仍然是匿名的。如果签名者反悔他的投标，那么无需签名者的合作，他的身份能被计算出。

下面描述一个群数字签名协议，该协议是由 Chaum 和 Van Heijst 于 1991 年提出的[8]。首先来描述该协议的基本思想。假定有 n 个人构成了一个群体 G，T 是一个可信中心，可信中心给每个成员分发一张秘密密钥表(这些表是互不相交的)，并且将该群体中所有成员的所有秘密密钥对应的公钥以随机的次序排成一张表公开。这样，每个成员就能用他自己的秘密密钥表中的秘密密钥签署消息，并且接收者能用公开的表中的公钥验证签名。每个密钥只能使用一次，否则，接收者能将这些签名联系在一起。因为是可信中心 T 产生的秘密密钥表，所以 T 知道公钥和签名成员之间的对应关系，一旦发生争端，T 就可解决这一争端。在这个方案中，每个成员得到一张固定的表，方案中公钥的总数是群成员人数 n 的一个线性函数，但每个成员所能签名的次数是固定的。

设 p 是一个使得在 Z_p 上计算离散对数是不可行的大素数，g 是 Z_p 的一个生成元。设群中共有 n 个成员，每一个群成员只有一个基本的秘密密钥 $s_i(1\leqslant i\leqslant n)$，对应的公钥是 $g^{s_i}\bmod p(1\leqslant i\leqslant n)$，可信中心 T 有一张这些公钥和群成员的名字相对应的表。可信中心

T 对每一个群成员 i 随机选择一个数 $r_i \in Z_p^*$（这里的 r_i 也称为盲因子），并将表 $(g^{s_i})^{r_i}$（$1 \leqslant i \leqslant n$）公开（这种公钥也称为盲公钥）。每个群成员 i 可使用 $s_i r_i \bmod (p-1)$ 作为秘密密钥，使用 ElGamal 型数字签名算法、不可否认的数字签名协议等对消息 x 进行签名。可信中心 T 可周期性地给群成员更换盲因子 r_i（$1 \leqslant i \leqslant n$）。

该协议的签名过程和验证过程与所选用的数字签名协议有关。当接收者收到一个消息的签名时，他试着用公开的公钥表中的每一个公钥去验证，只要有一个公钥使签名通过验证，就说明这个签名是该群的一个合法签名。如果发生争端，接收者可将通过验证的公钥和签名一起送交给可信中心 T，T 可利用这些信息及他存储的公钥与群成员名字的对应表找出对应的签名者。

8.5　盲数字签名协议

盲数字签名[9]在某些参加者需要匿名的安全协议中有着广泛而重要的应用，如在选举协议、安全的电子支付系统中。盲数字签名协议是满足下列两项要求的一种数字签名协议：①消息的内容对签名者是盲的（不可见的）；②在接收者将盲签名转化为非盲签名后，签名者不能追踪签名。目前已有大量文献讨论了盲数字签名协议的实现和应用问题。本节介绍两种类型的盲数字签名协议，它们分别基于 RSA 密码算法和离散对数问题。

做盲数字签名时，接收者先将要求签名的消息进行盲变换，把变换后的消息（称为盲消息）发送给签名者，签名者对盲消息进行签名并把签名送还给接收者，接收者对签名再做逆盲变换，得出的消息即为原消息的盲签名。这个过程如图 8.1 所示。

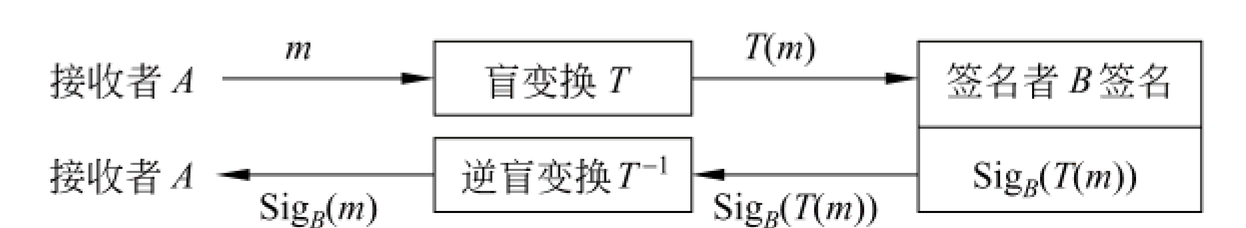


图 8.1　盲数字签名协议的签名过程

盲变换实现了盲数字签名的上述两项要求。下面首先来描述一个基于 RSA 密码算法的盲数字签名协议。

签名者 B 选择两个大素数 p 和 q，以及一个单向函数 f，并随机选择一个 e，使得 $\gcd(e,\varphi(n))=1$，其中 $n=pq$。由 $ed \equiv 1(\bmod \varphi(n))$ 求出 $d \equiv e^{-1} \bmod \varphi(n)$。$B$ 公开 (n,e) 和 f，保密 p、q 和 d。

如果接收者 A 想让 B 给他盲签一个消息 x，那么 A 先随机选择一个数 $k \in Z_p^*$（k 称为盲因子），计算 $x' = f(x) k^e \bmod n$，并将 x' 发送给 B。B 收到 x' 进行签名得 $y' = \mathrm{Sig_B}(x') = (x')^d \bmod n$。然后 B 将签名 y' 发送给 A，A 计算 $y = \dfrac{y'}{k} = f^d(x) \bmod n$。现在 A 得到了 B 对 x 的一个盲签名 y。显然，y 是 x 的一个合法签名。但 B 在签名过程中从来没有看到过 x 和 y，他无法将 (x,y) 和 (x',y') 联系起来。

其次，描述一个基于离散对数问题的盲数字签名协议。

签名者选择两个大素数 p 和 q，使得在 Z_p 上计算离散对数是困难的，并且 $q \mid (p-1)$。选择一个阶为 q 的数 $\alpha \in Z_p^*$ 和一个秘密密钥 x，公开 p、q、α 和 $y = \alpha^x \bmod p$，保密 x。

如果接收者 A 想让签名者 B 给他盲签一个消息 m，那么 B 先随机选择一个数$\tilde{k}\in Z_q^*$，计算$\tilde{r}=\alpha^{\tilde{k}} \bmod p$，并将$\tilde{r}$发送给 A。A 随机选择两个数 $a,b\in_R Z_q$，计算$r=\tilde{r}^a\alpha^b \bmod p$，$\tilde{m}=am\tilde{r}r^{-1} \bmod q$，并将$\tilde{m}$发送给 B。B 计算$\tilde{s}=(x\tilde{r}+\tilde{k}\tilde{m}) \bmod q$，并将 $\tilde{s}$ 发送给 A。A 计算 $s=(\tilde{s}r\tilde{r}^{-1}+bm) \bmod q$，则 B 对消息 m 的盲签名是整数对 (r,s)，验证方程是 $\alpha^s=y^r r^m \bmod p$。

发送者 A		签名者 B
		$\tilde{k}\in_R Z_q^*$
	$\xleftarrow{\tilde{r}}$	$\tilde{r}=\alpha^{\tilde{k}} \bmod p$
$a,b\in_R Z_q$		
$r=\tilde{r}^a\alpha^b \bmod p$		
$\tilde{m}=am\tilde{r}r^{-1} \bmod q$	$\xrightarrow{\tilde{m}}$	$\tilde{s}=(x\tilde{r}+\tilde{k}\tilde{m}) \bmod q$
$s=(\tilde{s}r\tilde{r}^{-1}+bm) \bmod q$	$\xleftarrow{\tilde{s}}$	

盲数字签名在某种程度上保护了用户的利益，但盲数字签名的匿名性也能被犯罪分子滥用，为了阻止这种滥用，人们引入了公平盲数字签名的概念并给出了一些具体实现。公平盲数字签名比盲数字签名多了一个特性：借助可信中心可将消息-盲签名对和签名者在执行签名协议时观察到的对应的盲消息联系起来。也就是说，通过可信中心，签名者可追踪签名。

8.6 门限数字签名协议

门限密码技术最早是由 Desmedt 和 Frankel 提出的[10,11]，是以秘密共享技术为基础的一种技术。文献[12]和[13]分别考虑了门限 ElGamal 密码技术和门限 DSA 密码技术。其基本思想是将加密或者签名所需要的密钥通过秘密共享技术分散到多个系统部件中，只有当足够数量（这个数量称为门限数量）的部件联合，才能完成一次加密和签名功能。换言之，只有获得了门限数量部件的控制权，才可能恢复出系统密钥；只要有门限数量的系统部件正常工作，系统就可以正常工作，并且系统就可以采取措施恢复到最初的安全状态。

下面介绍文献[13]中提出的门限 DSA 签名协议。假设系统中有 n 个参与者 $P_1,P_2,\cdots,P_n$，对消息 x 进行门限签名。

首先，介绍协议中使用的联合 Shamir 随机秘密共享（简称 Joint_Shamir_RSS）协议，分布式计算两个共享秘密乘积的分享的协议和分布式计算模逆的分享的协议。

Joint_Shamir_RSS 协议如下：

(1) 每个参与者 P_i 在 Z_q 上随机选取一个 t 次多项式 $f_i(x)(t+1<n)$，使得$s_i=f_i(0)$为 P_i 的秘密信息，P_i 给每个参与者 P_j 分发秘密分享 $\sigma_{ij}=f_i(j)$。

(2) 每个参与者 P_j 计算他的秘密分享信息$\sigma_j=\sum_{i=1}^{n}\sigma_{ij}$（当 P_j 没有收到从 P_i 发来的信息时，令 $\sigma_{ij}=0$），那么就得到$(\sigma_1,\cdots,\sigma_n)\overset{(t,n)}{\leftrightarrow}\sigma \bmod q$，其中$(\sigma_1,\cdots,\sigma_n)\overset{(t,n)}{\leftrightarrow}\sigma \bmod q$ 表示由

Joint_Shamir_RSS 协议生成的分享序列$\{\sigma_i\}_{i=1}^n$，其秘密共享信息是$\sigma=\sum_{i=1}^n f_i(0)$，而且该序列构成一个$(t+1,n)$门限序列。

给定两个共享秘密 u 和 v，如何在不揭示 u 和 v 的任何信息的情况下计算出秘密信息 $u\cdot v$ 的秘密分享？最直接的做法是：每个参与者 P_i都将对应于秘密 u 和v 的分享 σ_i 和 ρ_i 进行相乘，他将收到一个次数为 $2t$ 的多项式上的 $u\cdot v$ 的秘密分享，秘密信息可由 $2t+1$ 个正确的秘密分享重构出。文献[13]中举例说明了这样做的随机化程度不够好，为此，提出了以下的分布式计算两个共享秘密乘积的分享的协议。

(1) 采用 $2t$ 次的多项式，利用 Joint_Shamir_RSS 协议生成零秘密的分享序列，即对所有的 $i=1,2,\cdots,n$，$f_i(x)$的次数为 $2t$ 且 $f_i(0)=0$。

(2) 将 u 和 v 的秘密分享相乘，并与(1)中产生的零秘密的分享相加。

容易看到，将零秘密的分享相加到一个实际秘密的分享上，只是随机化了分享并没有改变实际秘密。

给定共享秘密 $k \bmod q$ 的秘密分享，如何在不揭示 k 和 k^{-1}的情况下计算出$k^{-1} \bmod q$ 的秘密分享？其基本思想是：联合生成一个秘密信息 b，公开值 $b\cdot k$，然后计算$(b\cdot k)^{-1}$并从所得结果中找到 k^{-1}的秘密分享。分布式计算模逆的分享的协议如下。

(1) 利用 Joint_Shamir_RSS 协议生成$(b_1,\cdots,b_n)\overset{(t,n)}{\leftrightarrow} b \bmod q$，其中 b_i是 P_i保存的秘密分享信息。

(2) 利用分布式计算两个共享秘密乘积的分享的协议，计算出 $b\cdot k$ 的秘密分享。

(3) 公开 $b\cdot k$ 的秘密分享。

(4) P_i通过以下公式计算 k^{-1}的秘密分享 u_i：$u_i=(b\cdot k)^{-1}b_i \bmod q$。

在门限 DSA 签名协议中，需要计算$(\alpha^{k^{-1}} \bmod p) \bmod q$ 的秘密分享而不是k^{-1}的秘密分享。具体过程如下。

(1) 像上面一样计算出 $b\cdot k$ 的秘密分享。

(2) 每个参与者 P_i广播$\alpha^{b_i} \bmod p$。

(3) 对指数利用拉格朗日插值公式可计算出 $\alpha^b \bmod p$。

(4) 计算 $\gamma=(\alpha^b)^{(b\cdot k)^{-1}}=\alpha^{k^{-1}} \bmod p \bmod q$。

其次介绍具体的门限 DSA 签名协议，与标准 DSA 签名算法不同的是，该协议中把 k 换成了 k^{-1}，因此参数 $\gamma=(\alpha^{k^{-1}} \bmod p) \bmod q$。协议如下。

(1) 采用 t 次多项式，利用 Joint_Shamir_RSS 协议生成$(k_1,\cdots,k_n)\overset{(t,n)}{\leftrightarrow} k \bmod q$，其中 k_i是 P_i 保存的秘密分享信息，k 是所生成的共享的随机秘密。

(2) 像上面一样，计算 $\gamma=\alpha^{k^{-1}} \bmod p \bmod q$。

(3) 按以下方式计算 $\delta=k(x+a\gamma) \bmod q$($a$ 是秘密共享的签名私钥)：

① 像上面一样通过相乘两个共享秘密 a 和 k 计算和公开 δ 的秘密分享。

② 从秘密分享重构 δ。

(4) 输出 x 的 DSA 签名(γ,δ)。

验证签名比较简单，只需要检查 $\gamma=\alpha^{x\delta^{-1}}\beta^{\gamma\delta^{-1}} \bmod p$ 是否成立即可。

参数 n 和 t 之间的关系与协议的抗攻击能力有关，这里就不再讨论了，感兴趣的读者可

参阅文献[13]。

8.7 存在特权集的门限数字签名协议

门限数字签名协议就是签名的责任被一个由多个签名方组成的签名方集合分享，其基本思想是：n个签名方中，至少要有t(不大于n)个签名才能产生一份消息的合法签名。门限数字签名协议的一个潜在问题是：各签名方的权限是等同的，但实际情形并不总是这样。我们在2000年提出了一个新型的门限数字签名模型[14]，这种模型的一个简单示例是：一个公司包含两类董事，一类是任职董事(8人)，另一类不在公司任职董事(12人)；要通过一个提案时，除需半数以上董事同意(对提案签名)外，为保证可操作性，还要求其中至少包括6名任职董事，同时要保证表决的匿名性，但事后在得到授权时应有办法查明表决情况。

更一般地，以上情形可以抽象成一个“存在特权集的门限数字签名模型”：一个由n个签名方组成的群体G，有m个不相交特权子集$G_1,\cdots,G_m$，每个子集有n_i人；要产生对某消息的合法群签名，至少需要$G_i(i=1,2,\cdots,m)$中有t_i人同意，且同意签名方总人数至少为t，还要求具有匿名性和事后确认签名方身份等性质。该模型可称为$(t_1,n_1;\cdots,t_m,n_m;t,n)$门限数字签名模型。实际上该模型是已有相关模型的一般化。

下面给出实现上述模型的一个具体协议[15]。

为了叙述方便起见，不妨假设签名方群体G只有一个特权子集G_1，下面将会看到，推广到多个特权子集的情况是很容易的。对应的存在特权集的门限数字签名协议记为$(t_1,n_1;t,n)$门限数字签名协议。

引入以下记号：

KAC：可信密钥认证中心，负责颁发密钥。

SC：签名服务机构，负责颁布签名。

G：n个签名方组成的群体。

G_1：G的子集，至少有其中的t_1方参与才可能产生合法签名，称为特权子集。

KAC首先选取安全素数p、q，满足$q|(p-1)$；其次在Z_q上秘密随机选取2个多项式$f(x)$，$g(x)$，次数分别为$(t-1)$和(t_1-1)；最后选取Z_q的本原元α，公开(p,q,α)和x_i，$y_{1j}\in_R Z_q$，$i=1,2,\cdots,n$，$j=1,2,\cdots,n_1$。

1. *群密钥和群私钥秘密分享的产生*

记Shamir秘密共享协议为SSS。

群私钥：由KAC随机产生，亦即$(f(0)+g(0)) \bmod q$。

群公钥：$z=\alpha^{(f(0)+g(0)) \bmod q} \bmod p$。

群私钥秘密分享分发：采用“双重”SSS(分别为(t,n)和(t_1,n_1)门限SSS)，即

(1) 如果i是普通用户，则得到对应的群私钥秘密分享为$f(x_i)$，并由KAC公开其公钥$z_i=\alpha^{\lambda_i f(x_i)} \bmod p$。

(2) 如果i是特权集中的用户(简称为特权用户)，则得到对应的群私钥秘密分享为$f(x_i)$，$g(y_{1i_j})$，公开$z_i=\alpha^{\lambda_i f(x_i)+\mu_i g(y_{1i_j})} \bmod p$，这里$\lambda_i$、$\mu_i$是SSS所涉及的拉格朗日恢复系数，是公开可计算的。以上过程实际上是建立了各用户的公、私钥秘密分享。

2. $(t_1, n_1; t, n)$门限数字签名的产生

不妨假设只有 t 个人参加签名，且恰为 $1,2,\cdots,t$，设被签署的消息为 m。具体步骤如下。

(1) 单个签名的产生和验证。对每个 $i=1,2,\cdots,t$，i 秘密随机选取 $k_i \in Z_p^*$，计算 $r_i = \alpha^{k_i} \bmod p$，并在群内通过广播信道匿名广播 r_i，于是，每个用户 i 都可以通过下式计算，即

$$r = \prod_{i=1}^{t} r_i \bmod p$$

若 i 是普通用户，则计算 $s_i = (f(x_i)\lambda_i h(m) - k_i r) \bmod q$；若 i 是特权用户，则计算 $s_i = (f(x_i)\lambda_i h(m) + g(y_{1i_j})\mu_i h(m) - k_i r) \bmod q$。这里 λ_i、μ_i 是 SSS 所涉及的拉格朗日恢复系数，是公开可计算的，$h(\cdot)$是安全的 Hash 函数。子签名(r, s_i)被发送给签名服务机构 SC。SC 可以按照以下办法验证所提交的子签名的合法性：对于用户i(无论是普通用户还是特权用户)，验证方程

$$\alpha^{s_i} r_i^r = z_i^{h(m)}$$

是否成立，若成立则接受子签名。

(2) 签名合成。如果 SC 接受所有提交的子签名，则计算 $s = (s_1 + s_2 + \cdots + s_t) \bmod q$，输出$(r, s)$作为对应消息 m 的门限签名。

3. 签名的验证和事后身份追踪

不妨仍假设共有 t 个人参加签名，且恰为 $1,2,\cdots,t$，其中至少有 t_1 人属于特权子集。显然有

$$\begin{aligned} s &= h(m)\left(\sum_{i=1}^{t} f(x_i)\lambda_i + \sum_{i=1}^{t_1} g(y_{1i_j})\mu_i\right) - r\sum_{i=1}^{t} k_i \\ &= h(m)(f(0) + g(0)) - r\sum_{i=1}^{t} k_i \end{aligned}$$

因此有以下验证方程成立，即

$$\alpha^s r^r = z^{h(m)}$$

易见，如果不符合特权条件要求，则即使有 t 个以上人员参加签名，分量 $g(0)$的恢复也是不可能的，从而得不到群私钥的任何信息；而如果不足 t 个以上人员参加签名，即使 $g(0)$ 可以恢复，但却不可能恢复 $f(0)$，还是得不到群私钥的任何信息。

如果事后得到许可，需要调查是哪些人员参与签名，则由 SC 追踪签名方是平凡的。

值得一提的是，$(t_1, n_1; t, n)$门限签名协议易于推广到$(t_1, n_1; \cdots, t_m, n_m; t, n)$门限签名协议。只需在协议建立阶段选取 $m+1$ 个多项式 $f(x), g_1(x), \cdots, g_m(x)$，群私钥被选取为 $\left(\sum_{i=1}^{m} g_i(0) + f(0)\right) \bmod q$，每个特权用户持有 $f(x_i)$及对应的 $g_i(y_{ij})$，其余同前。

以上协议也可以不要求 KAC 存在。基本方法是：各用户自己做自己的 KAC，即自行选择(如 ElGamal 型数字签名)公、私钥对(x_i, y_i)；群公钥为 $y = \prod_{i=1}^{n} y_i$，类似前面的分享分发过程，各用户采用“双重 SSS”方法，把自己的私钥分享分发给其余$(n-1)$个用户，门限签名的产生是类似的。实际上就是采用不存在可信中心的秘密共享协议为基本模块，建议采用 Pedersen 或 Feldman VSS 协议模块。

利用上述设计思想可以构造出具有消息恢复性质的($t_1,n_1;\cdots,t_m,n_m;t,n$)门限签名协议,也可以构造出存在特权集的代理门限签名协议。

8.8 可验证的签名共享协议

为了保护数字签名,Franklin 等人在文献[16]中提出了一种称之为可验证的签名共享(VSS)协议。这种协议的基本思想是:可使一个数字签名文件的拥有者(拥有者可以是也可以不是原来的签名者)把这个签名分配到一个所谓的代理集上,使得诚实的代理日后可以重构它。在共享阶段结束,每个代理能验证是否文件的一个合法签名能被重构,即使原来的签名拥有者或某些代理是有故障的。另外,在重构之前有故障的代理没有获得一个诚实的分发者拥有的关于签名的任何信息(但确实看到了文件本身)。文献[16]中基于 RSA 数字签名算法、ElGamal 数字签名算法、Schnorr 数字签名算法和 DSA 等给出了一些可验证的签名共享协议。但文献[17]中指出基于 DSA 的 VSS 协议是不安全的,并提出了两个新的基于 DSA 的 VSS 协议。

一个 VSS 协议由共享协议和重构协议组成,参与者包括一个分发者 D、一个重构者 R 和 n 个代理 $P_1,P_2,\cdots,P_n$。这里假定 R 是诚实的。

我们说一个 VSS 协议是 t 弹性的,如果它满足以下条件:

(1) 完全性(Completeness)。如果 D 是诚实的,至多 t 个代理是有故障的,则每个诚实的代理都将接受。

(2) 合理性(Soundness)。如果至多 t 个代理是有故障的,并且任何诚实的代理在共享阶段结束后接受,则每个诚实的代理在共享阶段结束后接受并可被成功地重构。

(3) 秘密性(Secrecy)。控制至多 t 个有故障代理的敌手在参加 k 次具有诚实的 D 的共享协议之后所能计算的东西,与他没有参加任何共享协议所能计算的东西一样多。

我们说一个 VSS 协议是具有启发式秘密 t 弹性的,如果该协议的完全性和合理性是可证明的,而秘密性是启发式的,即有一些证据表明敌手似乎不能获得任何有用的信息。

8.8.1 可验证的离散对数共享协议

一个公开值的可验证的离散对数共享(Verifiable Discrete Log Sharing)协议本质上是一个可验证的秘密共享协议(当然可能在模型上稍有差异),它是 VSS 协议设计的核心。

设 p、q、g 是公开的,p 和 q 都是大素数且 $q\mid(p-1)$,g 是 Z_p^* 中的一个 q 阶元素,令 $\langle g\rangle=\{1,g,\cdots,g^{q-1}\}$。分发者 D 广播 α 并共享一个值 β,他声称:①$\alpha\in\langle g\rangle$;②$\beta$ 是 α 关于基底 g 的离散对数,即 $\alpha=g^\beta \bmod p$。

一个可验证的离散对数共享协议由共享协议和重构协议组成,下面分别介绍这两个协议。

1. 共享协议

(1) D 秘密地发送消息并可靠地广播。

① D 随机选择 $a_t,\cdots,a_1\in_R Z_q$,令 $f(x)=a_tx^t+\cdots+a_1x+\beta$。

② D 秘密地给 P_i 发送消息 $\beta_i=f(i)\bmod q$,$1\leqslant i\leqslant n$。

③ D 可靠地广播 $\alpha, g^{a_1} \bmod p, \cdots, g^{a_t} \bmod p$。

(2) 设从(1)的③中得到的广播值表示为 $\alpha, u_1, \cdots, u_t$。每个代理 P_i 对所有的代理做以下的可靠广播。

① 如果 $g^{\beta_i} \not\equiv \alpha \prod_{j=1}^{t} (u_j)^{i^j} (\bmod p)$，则 COMPLAIN(投诉)。

② 否则，ALLOW(允许)。

(3) 如果下列条件成立，则每个代理都 ACCEPTS(否则 REJECTS)。

① 至多 t 个代理 COMPLAINED(在(2)的①中)。

② $\alpha^q \equiv 1(\bmod p)$。

2. 重构协议

(1) 每个代理 P_i 发送给 R 如下信息。

① β_i。

② $\alpha, u_1, \cdots, u_t$。

(2) R 按以下方式恢复 α 关于基底 g 的离散对数。

① R 利用择多原则可从(1)的②中收到的值中找出 $\alpha, u_1, \cdots, u_t$。

② R 从(1)的①中收到的值中去掉那些使得 $\alpha, u_1, \cdots, u_t$ 不一致的，即 $g^{\beta_i} \not\equiv \alpha \prod_{j=1}^{t} (u_j)^{i^j} \bmod p$ 的秘密分享 β_i。

③ R 从(1)的①中收到的剩余秘密分享，利用拉格朗日插值公式恢复出一个次数至多为 t 的多项式 $f(x) \in Z_q[x]$。离散对数 $\beta = f(0)$。

8.8.2 Franklin 等人的基于 DSA 的 VSS 协议及其安全性分析

DSA 的描述参见 1.3.2 小节，这里也采用 1.3.2 小节的参数和符号。

Franklin 等人的基于 DSA 的 VSS 协议的工作流程如下。

(1) 分发者 D 将 x、δ、s 可靠地分发给所有的代理，其中 x 是被签名的文件或消息，$\delta = (\text{SHA}-1(x) + a\gamma) k^{-1} \bmod q$ 是 x 的签名的一部分，$s = \beta^{e_2} \bmod p$，$e_2 = \gamma\delta^{-1} \bmod q$，$\gamma = (\alpha^k \bmod p) \bmod q$，$\beta = \alpha^a \bmod p$，$a$ 是签名者的秘密密钥，k 是签名者在签名消息 x 时秘密随机选择的一个数。

(2) D 对代理们做 e_2 的一个可验证的共享(可用 8.8.1 小节介绍的可验证的离散对数共享协议)使得他们相信 e_2 是 s 关于基底 β 的离散对数。

(3) 如果代理们接受对数共享协议并且 $\beta^t = s^\delta \bmod p$，$t = (\alpha^{e_1} s \bmod p) \bmod q$，$e_1 = \text{SHA}-1(x)\delta^{-1} \bmod q$，那么他们接受。签名的另一部分可由 $\gamma = e_2\delta \bmod q$ 构造出。

文献[16]中声称上述协议在一定假设下是安全的，但文献[17]中指出上述协议是不安全的。这是因为当一个代理知道 x、δ 时，e_1 可由公式 $e_1 = \text{SHA}-1(x)\delta^{-1} \bmod q$ 计算出，从而可计算出 $\alpha^{e_1} \bmod p$，又每个代理都知道 s，所以他可以计算出 $\gamma = (\alpha^{e_1} s \bmod p) \bmod q$。可见，无须知道 e_2 即 s 关于基底 β 的离散对数就能恢复 γ，这表明收到分发值 x、δ、s 的任何代理都能重构 γ。实际上 $\gamma = t$。因此，上述协议不安全。

8.8.3 两个基于 DSA 的 VSS 协议

文献[17]在文献[16]的基础上，提出了两个基于 DSA 的 VSS 协议。当 $n \geq 3t+1$ 时，

这些协议是具有启发式秘密 t 弹性的。

1. 第一个协议的工作流程

(1) 分发者 D 将 x、γ、s、A 可靠地分发给所有的代理，其中 x 是被签名的文件或消息，$\gamma=(\alpha^k \bmod p) \bmod q$ 是 x 签名的一部分，$s=\beta^{e_2} \bmod p$，$A=s^{\delta} \bmod p$，$e_2=\gamma\delta^{-1} \bmod q$，$\delta=(\text{SHA}-1(x)+a\gamma)k^{-1} \bmod q$，$\beta=\alpha^a \bmod p$，$a$ 是签名者的秘密密钥，k 是签名者在签名消息 x 时秘密随机选择的一个数。

(2) D 对代理们做 δ 的一个可验证的共享(可用 8.8.1 小节介绍的可验证的离散对数共享协议)使得他们相信 δ 是 A 关于基底 s 的离散对数。

(3) 如果代理们接受对数共享协议并且 $A=\beta^{\gamma} \bmod p$，那么他们接受。签名的另一部分 δ 可从对数共享协议中重构。

2. 第二个协议的工作流程

(1) 分发者 D 将 x、s_1、s_2 可靠地分发给所有的代理，其中 x 是被签名的文件或消息，$s_1=\alpha^{e_1} \bmod p$，$s_2=\beta^{e_2} \bmod p$，$e_1=\text{SHA}-1(x)\delta^{-1} \bmod q$，$e_2=\gamma\delta^{-1} \bmod q$，$\delta=(\text{SHA}-1(x)+a\gamma)k^{-1} \bmod q$，$\gamma=(\alpha^k \bmod p) \bmod q$，$\beta=\alpha^a \bmod p$，$a$ 是签名者的秘密密钥，k 是签名者在签名消息 x 时秘密随机选择的一个数。

(2) D 对代理们做 e_2 的一个可验证的共享(可用 8.8.1 小节介绍的可验证的离散对数共享协议)使得他们相信 e_2 是 s_2 关于基底 β 的离散对数。

(3) 如果代理们接受对数共享协议并且 $s_1^{\gamma} \equiv s_2^{\text{SHA}-1(x)} \pmod p$，那么他们接受。其中 γ 可由公式 $\gamma=(s_1 s_2 \bmod p) \bmod q$ 获得，签名的另一部分 δ 可由 $\delta=e_2\gamma \bmod q$ 构造出。

在上述两个协议中，给定 x、γ、s、A 或 x、s_1、s_2 要计算出 δ，没有计算离散对数能力的人似乎是不可能的，从这里可以推断这些协议具有秘密性。关于这些协议的其他性质如完全性、合理性等的讨论类似于文献[16]中的有关讨论，这里就不再赘述。

8.9 门限签密协议

现实世界中存在以下这种情况：团体 A 要发送某消息给团体 B，要求仅当 A 中特定数目以上的人联合签名消息才能生效，而且仅当 B 中一定数目以上的人合作才能正确阅读并认证该消息，如公司间的商务往来等，一般还可以要求满足非否认性(Non-Repudiation)。把这种模型称为“门限签密模型”。

简单地复合使用门限加密技术和门限签名技术，可以给出实现上述模型的一个具体协议，但显然实现代价较高，而且可能要重复执行许多步骤。因此，必须寻找节省资源(如带宽)、快速有效的实现途径。

8.9.1 改进的基础签密算法

为了给出一个实现“门限签密模型”的有效协议，我们对文献[18]中的签密算法进行了修改，得到了以下的基础签密算法 $\text{SC}=(\text{Com}(k), K_{\text{A}}(k, \text{cp}_{\text{sc}}), K_{\text{B}}(k, \text{cp}_{\text{sc}}), \text{SC}^{G,H}_{x_{\text{A}},y_{\text{B}}}(m), \text{USC}^{G,H}_{y_{\text{A}},x_{\text{B}}}(C))$。

(1) $\text{Com}(k)$(用于产生公共参数)：随机选择两个大素数 p 和 q，$|p|=k$(安全参数)，

$q|(p-1)$，$q>2^{l_q(k)}$；$g\in_R Z_p^*$，$\mathrm{Ord}(g)=q$，输出 $cp_{sc}=(p,q,g)$（符号 $\in_R$ 表示随机取自）。

(2) $K_A(k,cp_{sc})$（用于产生发送方公、私钥）：$x_A\in_R Z_q$，$y_A=g^{x_A}\bmod p$，输出(y_A,x_A)。

(3) $K_B(k,cp_{sc})$（用于产生接收方公、私钥）：$x_B\in_R Z_q$，$y_B=g^{x_B}\bmod p$，输出(y_B,x_B)。

(4) $SC_{x_A,y_B}^{G,H}(m)$（对消息 m 签密）：$x\in_R Z_q$，$K=y_B^x \bmod p$，$\tau=G(K)$。$\mathrm{bind}=y_A\|y_B$，$c=E_\tau(m)$，$r=H(m\|\mathrm{bind}\|K)$，$s=(rx_A-x)\bmod q$，输出签密 $C=(c,r,s)$；其中，G、H 是安全的 Hash 函数，E_τ 是以 τ 为密钥的抵抗选择明文攻击安全的对称加密算法，如 AES。

(5) $USC_{y_A,x_B}^{G,H}(C)$（收方解密认证）：若签密 C 的 3 个分量中有一个不在正确的定义域，拒绝；否则计算 $w=y_A^r g^{-s}\bmod p$，$K=w^{x_B}\bmod p$，$\tau=G(K)$，解密 $m=D_\tau(c)$，若满足 $H(m\|\mathrm{bind}\|K)=r$，则返回消息 m，否则拒绝。

上述签密算法与文献[18]中的签密算法相比，只修改了签密 C 的第三个分量s 的计算方法，即把原来的公式 $s=x/(r+x_A)\bmod q$ 改为 $s=(rx_A-x)\bmod q$，显然不是本质区别，主要目的是方便 8.9.2 小节的门限签密协议的设计。不难看出，签密方案较传统标准方法（如 Signature then Encryption）在节省计算资源、提高效率等方面有很大优势。

8.9.2 门限签密协议

本节介绍的门限签密协议来源于文献[19]，该门限签密协议主要由以下 3 个协议组成：密钥生成协议、$(t+1,n)$门限签密协议和$(t'+1,m)$门限解密验证协议。

1. 密钥生成协议

采用文献[20]中提出的一种可证明安全的密钥分配协议 DKG 作为密钥生成协议。在该协议中，有 n 个用户，记为 $P_1,P_2,\cdots,P_n$。该协议在没有任何可信中心的情况下协同生成群公、私钥。

设 g、h 是 2 个阶为 q 的生成元（假定离散对数 $\log_g h$ 是难处理的）。产生私钥x 的过程如下。

(1) P_i作为自己的可信中心利用 Pedersen 可验证的秘密共享协议产生随机值 z_i：

① P_i 选 2 个次数为 t 的随机多项式 $f_i(z)$，$f_i'(z)\in Z_q[z]$，

$$f_i(z)=a_{i0}+a_{i1}z+\cdots+a_{it}z^t,\quad f'_i(z)=b_{i0}+b_{i1}z+\cdots+b_{it}z^t$$

设 $z_i=a_{i0}=f_i(0)$，P_i广播 $C_{ik}=g^{a_{ik}}h^{b_{ik}}\bmod p\ (k=0,1,\cdots,t)$，计算 $s_{ij}=f_i(j)$，$s_{ij}'=f_i'(j)\ (j=1,2,\cdots,n)$，秘密地发送给每个 P_j。

② $P_j(j=1,2,\cdots,n)$验证等式

$$g^{s_{ij}}h^{s'_{ij}}\equiv\prod_{k=0}^{t}(C_{ik})^{j^k}\ (\mathrm{mod}\ p)\tag{8-2}$$

是否成立，如果失败则广播对 P_i 的一个投诉（Complaint）。

③ 作为可信中心的 P_i，在收到一个来自 P_j 的投诉后，广播满足等式(8-2)的 s_{ij}，s_{ij}'。

④ 如果以下任何一种情况发生，P_i 均被标记为不合格（Disqualified）：在②中受到 t 个以上投诉或在③中回答投诉时，仍用不满足等式(8-2)的错误值回答。

(2) 每一个 P_i 建立合格用户集合 QUAL。

(3) 群私钥为 $x=\sum_{i\in \mathrm{QUAL}}z_i\bmod q$，每一个 P_i 的相应分享为

$$x_i=\sum_{j\in \mathrm{QUAL}}s_{ji}\bmod q,\quad x'_i=\sum_{j\in \mathrm{QUAL}}s'_{ji}\bmod q$$

群公钥 $y=g^x \bmod p$ 的生成过程如下。

(1) 利用 Fedleman 可验证的秘密共享协议揭示 g^{z_i} ($y_i=g^{x_i} \bmod p$ 实际上也以隐含方式揭示):

① P_i(i 属于 QUAL)广播 $A_{ik}=g^{a_{ik}} \bmod p(k=0,1,\cdots,t)$。

② 每一 P_j 验证:对于 i 属于 QUAL 等式

$$g^{s_{ij}} \equiv \prod_{k=0}^{t}(A_{ik})^{j^k} (\bmod p) \tag{8-3}$$

是否成立。若不成立,通过广播满足等式(8-2)但不满足式(8-3)的 s_{ij} (s'_{ij})来投诉P_i(注意等式(8-3)成立实际上等价于公开了 $g^{s_{ij}} \bmod p$)。

③ 对于在②中至少受到一个合法投诉的 P_i,其他合法用户运行 Pedersen 可验证的秘密共享协议的恢复算法,从而计算出(z_i,$f_i(z)$,A_{ik})($k=0,1,\cdots,t$)。任意 QUAL 中用户具有子公钥

$$y_i = g^{\sum_{j\in \mathrm{QUAL}} s_{ji}} = g^{x_i} \bmod p$$

(2) 根据等式(8-3),任何一个 QUAL 中用户都可以计算 y_i,因此各方得到群公钥

$$y = \prod_{i\in \mathrm{QUAL}} g^{z_i} \bmod p = \prod_{i\in \mathrm{QUAL}} A_{i0} \left(= \prod_{i\in \mathrm{QUAL}} y_i^{\lambda_i} \bmod p\right)$$

这里 λ_i 是对应的拉格朗日插值多项式系数,是公开可计算的。

发送方 $A=\{A_1,\cdots,A_n\}$和接收方 $B=\{B_1,\cdots,B_m\}$分别调用($t+1$)-门限 DKG 协议和($t'+1$)-门限 DKG 协议可以生成。

(1) 发送方群公钥 $y_A=g^{x_A}$,子公钥 $y_{Ai}=g^{x_{Ai}}$ ($i=1,2,\cdots,n$)。

(2) 接收方群公钥 $y_B=g^{x_B}$,子公钥 $y_{Bi}=g^{x_{Bi}}$ ($i=1,2,\cdots,m$)。

这里 x_{Ai}和 x_{Bi}分别对应 A_i和 B_i持有的秘密分享。

2. ($t+1,n$)门限签密(TSC)协议

公开参数的产生类似于基础签密算法 SC。开始执行协议时所有参与方都被标记为合格(Qualified)。具体过程如下:

(1) 每个 $A_i \in$ QUAL 随机选取 $x_i \in Z_q$,公开承诺 $X_i=g^{x_i} \bmod p$。

(2) 每个 A_i 计算并广播 $y_B^{x_i}$。

(3) 每个 A_i 根据广播值计算

$$K = \prod_{i\in \mathrm{QUAL}} (y_B^{x_i})^{\lambda_i} \bmod p(\text{形式记为 } y_B^{x}),$$

$$\tau = G(K), \quad c = E_\tau(m), \quad r = H(m \,\|\, \mathrm{bind} \,\|\, K),$$

这里 bind$=y_A \| y_B$(λ_i 是相应的拉格朗日系数,是公开可计算的)。

(4) 每个 A_i 计算并广播 $s_i=(rx_{A_i}-x_i) \bmod q$。

(5) A_j($j\in$QUAL)验证等式

$$g^{s_i}X_i = y_{Ai}^r \tag{8-4}$$

是否成立,若不成立,投诉 A_i。

(6) A_i 若收到 1 个以上合法投诉,则标记为不合格,其余各合法方返回到(1)。

(7) 如果对所有 QUAL 中用户而言,等式(8-4)都成立,即最终确立 QUAL。

这时任意 QUAL 中用户 A_i(或任意一个组合器 Combiner)计算

$$s=\sum_{i\in \mathrm{QUAL}}\lambda_i s_i\left(=rx_{\mathrm{A}}-x=\left(rx_{\mathrm{A}}-\sum_{i\in \mathrm{QUAL}}\lambda_i x_i\right)\bmod q\right)$$

输出 $C=(c,r,s)$作为门限签密。

注：(1)、(2)均可以预先离线计算。

3. $(t'+1,m)$门限解密验证(TUSC)协议

该协议是由接收方 $B=\{B_1,\cdots,B_m\}$联合执行的。具体过程如下。

(1) 每个 B_i 把签密划分为 $C=c\parallel r\parallel s$，检查各分量是否都在正确的定义域内，若不是，拒绝接收。

(2) 每个 B_i 计算 $w=y_{\mathrm{A}}^{r}g^{-s}\bmod p$。

(3) 每个 B_i 计算并广播$(w^{x_{\mathrm{B}i}},pf)$，这里 $pf=(pf_1,pf_2)$，是通过以下计算给出的：令 $z=w^{x_{\mathrm{B}i}}\bmod p$；$l\in_R Z_q$，$u=g^l$，$v=w^l$。

$pf_2=H'(g,w,y_{\mathrm{B}i},z,u,v)$，$pf_1=(l+pf_2 x_{\mathrm{B}i})\bmod q$($H'$是一个 Hash 函数)。

(4) 每个 B_j 通过以下方式验证 B_i 的广播值$(w^{x_{\mathrm{B}i}},pf)$：

$$u=g^{pf_1}y_{\mathrm{B}i}^{-pf_2}\bmod p,\quad v=w^{pf_1}(w^{x_{\mathrm{B}i}})^{-pf_2}\bmod p,\quad c'=H'(g,w,y_{\mathrm{B}i},w^{x_{\mathrm{B}i}},u,v)$$

如果 $c'=pf_2$，则认为正确，否则产生对 B_i的一个投诉。

(5) 如果对 B_i 至少存在 1 个合法投诉，B_i 被标记为不合格，设合格用户集 QUAL。

(6) $K=\prod_{i\in \mathrm{QUAL}}w^{\mu_i x_{\mathrm{B}i}}\bmod p$，这里 μ_i 是对应拉格朗日系数。

(7) 每个 B_i计算：$\tau=G(K)$，$m=D_\tau(c)$；如果 $H(m\parallel \mathrm{bind}\parallel K)=r$，返回消息 m，否则拒绝。

8.10 指定验证方的签名协议

指定验证方的签名协议是这样一种协议，签名方 A 可以向指定的验证方 B 证明签名的合法性，但是任何第三方则无法从签名判断消息的起源(如 A 的身份等)。在这种协议中，只有指定的验证方 B 可以验证签名的合法性，因此签名方的身份隐私得以有效保护。在电子投标这类应用中，指定验证方的签名协议具有非常现实的应用需求。

1996 年 Laih 等人提出了一类指定验证方的签名协议(Specified Verifiers Signature，简称为 SV 签名协议)[21]。但文献[22]中指出了文献[21]给出的协议存在严重安全缺陷：SVS(指定的验证服务器集合)中的 Clerk 可以自己验证几乎所有的签名。

8.10.1 Laih 的 SV 签名协议

Laih 的 SV 签名协议[21]由以下 3 个算法组成：密钥生成算法、多签名算法和验证算法。

1. 密钥生成算法(KeyGen)

设 $G_s=\{U_{s1},\cdots,U_{sn}\}$为签名方群体，群私钥、公钥对为 $\left(\sum_{i=1}^{n}s_i,\ Y_s=\prod_{i=1}^{n}g^{-s_i}\bmod p\right)$；$G_v=\{U_{v1},\cdots,U_{vm}\}$为验证方群体，群私钥、公钥对为 $\left(\sum_{i=1}^{m}v_i,\ Y_v=\prod_{i=1}^{m}g^{-v_i}\bmod p\right)$；每个群里有一 Clerk；单个用户的公私钥对应以上分量，即$U_{si}$$(i=1,2,\cdots,n)$的私钥、公钥对为$(s_i,g^{-s_i})$；

$U_{vi}(i=1,2,\cdots,m)$的私钥、公钥对为(v_i,g^{-v_i})。上述安全参数都由可信中心产生。

2. 多签名算法(Multisign)

设待签消息为m,签名步骤如下。

(1) 每个U_{si}随机选择$r_i\in Z_q$,计算$x_i=Y_v^{r_i}$,并发送给Clerk_s。

(2) Clerk_s 计算 $x=\prod_{i=1}^{n}x_i \bmod p$,在$G_s$中广播$x$。

(3) 每个U_{si}计算$e=h(x,m)$和$w_i=r_i+es_i \bmod q$,w_i被交给Clerk_s。

(4) Clerk_s 计算 $e=h(x,m)$,$w=\sum_{i=1}^{n}w_i \bmod q$,输出多签名$(e,w,m)$。

3. 验证算法(Verification)

G_v中成员协同计算过程如下。

(1) 每个U_{vj}计算$x_j=(g^wY_s^e)^{-v_j} \bmod p$,传送给$\text{Clerk}_v$。

(2) Clerk_v 计算 $x=\prod_{i=1}^{n}x_i \bmod p$,广播$x$。

(3) 每个U_{vj}验证$e=h(x,m)$是否成立。

文献[22]中指出,以上协议存在以下安全缺陷:Clerk_v只要在某个时间参与过一次验证协议,则以后他可以单独验证所有签名。理由是由验证算法 Verification 易于计算出:

$$g^{\sum_{i=1}^{n}s_i\sum_{i=1}^{m}v_i}=(xY_v^{-w})^e,\quad x=Y_v^wg^{-e\sum_{i=1}^{n}s_i\sum_{i=1}^{m}v_i}$$

因此,只要Clerk_v计算过一次x,就可得到固定值$g^{\sum_{i=1}^{n}s_i\sum_{i=1}^{m}v_i}=(xY_v^{-w})^e$,从而以后可以验证任一签名$(e,w,m)$是否合法。

不难看出,不仅Clerk_v,每个验证方都可以在一次验证成功后具有验证以后所有签名的能力(无须 Clerk 或其他验证方的配合),具体方法同上。

8.10.2 指定验证方的签名协议

本小节介绍利用文献[23]中的签名协议 EDL 的基本思想设计的一个指定验证方的签名协议 SV-EDL[24]。

SV-EDL 涉及两个独立的安全 Hash 函数:

$$H:\{0,1\}^*\rightarrow G=\{g^0,g^1,\cdots,g^{q-1}\},\quad H':G^8\rightarrow Z_q$$

设验证服务器 SVS 的群公、私钥分别为$y_v=g^{x_v} \bmod p$和x_v,g的阶$q\mid(p-1)$。

SV-EDL 签名协议描述如下。设待签名消息为m。

1. 密钥生成算法 KeyGen

签名方的私钥为$x\in_R Z_q$,公钥为$y=g^x \bmod p$。

2. 签名算法 Sign

(1) 随机选择$r\in_R\{0,1\}^{n_r}$,利用 Hash 函数H计算$h=H(m,r)$。

(2) 计算$z=h^x$(注意离散对数$DL_g(y)=DL_h(z)=x$)。

(3) 随机选择$k\in_R Z_q$,计算$u=g^k$,$v=h^k$。

(4) 随机选择$l\in_R Z_q$,计算$w=g^l$,$w'=y_v^l(=g^{lx_v})$。

(5) 利用 Hash 函数 H' 计算 $c=H'(g,h,y,z,u,v,w,w')$。

(6) 计算 $s=(k+cx) \bmod q$，输出 m 的签名 $\sigma=(z,r,s,w,c)$。

3. 验证算法 $\mathrm{Ver}_{\mathrm{SVS}}$

SVS 计算 $h=H(m,r)$，$u=g^s y^{-c}$，$v=h^s z^{-c}$，$w'=w^{r_v}$，最后验证等式 $c'=H'(g,h,y,z,u,v,w,w')=c$ 是否成立以确定签名的真伪。

8.11　环签名协议

Rivest、Shamir 和 Tauman 于 2001 年提出了"环签名"的概念[25]，其基本思想是：允许某特定群体的任何一方代表整个群体签名，但却无法具体追踪特定的签名方。在此基础上 Kudla[26] 进一步得出了以下结论：指定验证方的签名协议与一类特殊的 2 方环签名协议是等价的，并给出了系统的分析设计方法。值得注意的是，环签名与群签名有很多类似之处，但还是有本质的不同，如没有管理员。

文献[27]中提出了一个很有代表性的环签名协议，目前很多环签名协议都吸取了其基本设计思想。将这个环签名记为 RSS，可由以下一组算法定义。

(1) 参数建立(Setup)：给定某安全参数 l，设 p、q 是 2 个大素数，满足 $q\,|\,(p-1)$；设 G 是 Z_p^* 的 q 阶乘法群，g 是 G 的生成元；设 H 是一个从 $\{0,1\}^*$ 映到 Z_q 的安全 Hash 函数，公开参数 params=(p,q,g,H)。

(2) 密钥生成(KeyGen)：该算法以 params 为输入，输出用户私钥 $x\in_R Z_q^*$，公钥 $X=g^x \bmod p$。

(3) 环签名(RingSign)：输入消息 m、公钥列表 $R=(X_1,X_2,\cdots,X_n)$ 及签名私钥 x_{sig}(对应公钥 X_{sig} 是环 R 的第 sig 个公钥)，随机选择 $h_i,k\in_R Z_q^*$，$i=1,2,\cdots,n,i\neq \mathrm{sig}$，计算

$$z=g^k\prod_{i=1,i\neq \mathrm{sig}}^{n} X_i^{h_i} \bmod p$$

$$h=H(X_1,\cdots,X_n,m,z)$$

$$h_{\mathrm{sig}}=\Big(h-\sum_{i=1,i\neq \mathrm{sig}}^{n} h_i\Big) \bmod q$$

$$s=(k-x_{\mathrm{sig}}h_{\mathrm{sig}}) \bmod q$$

输出环签名 $\sigma=(s,h_1,h_2,\cdots,h_n)$。

(4) 环签名验证(RingVerify)：以收到的消息 m、公钥列表 $R=(X_1,X_2,\cdots,X_n)$、环签名 $\sigma=(s,h_1,h_2,\cdots,h_n)$ 为输入，计算

$$h=H(X_1,\cdots,X_n,m,g^s X_1^{h_1}\cdots X_n^{h_n} \bmod p)$$

$$h_1+h_2+\cdots+h_n=h \text{ 是否成立?}$$

如果等式成立，输出接受；否则拒绝。

不难看出，上述环签名 RSS 本质上是从 Schnorr 签名[28]演变而来，而文献[29]中已经证明 Schnorr 签名在离散对数问题难解的意义下是不可伪造的(抗适应性选择消息攻击)，因上述环签名的不可伪造性事实上可以归约为离散对数问题[26,27]，即如果 Schnorr 签名是不可伪造的，则上述环签名 RSS 也是不可伪造的。

8.12 并发签名协议

环签名允许群体中任何成员代表群体签名，但任何人都无法追踪群体中的具体签名方，注意不存在任何管理员。文献[26]中在研究 2 方公平签名交换协议时，以2 方环签名（签名群体只有 2 方）为基础，通过一方在产生签名分量时植入陷门（称为 Keystone）的方法，提出了所谓的“并发签名”（Concurrent Signature）。如果满足正确性、不可传递性、不可伪造性和公平性，则称一个并发签名协议是安全的。所谓不可传递性，是指签名的合法性是由用户 1 和用户 2 共同来验证，任何一方不具有向第三方证明签名合法性的能力（不揭示 Keystone 值 k 的情况下）。而正确性、不可伪造性和普通签名的定义是类似的。公平性表示：①只有产生 Keystone 的一方可以揭示 k；②一旦揭示 Keystone，所有满足不可传递性的签名就和用户身份绑定。

并发签名协议是一个由以下算法组成的数字签名协议。

(1) 参数建立（Setup）：输入安全参数 l、输出公开参数（包括公钥空间 PK、私钥空间 SK、消息空间 M、签名空间 S、Keystone 空间 K、Keystone 迹空间 F 及函数 KCommit：$K \to F$ 的描述）的概率算法。

(2) 密钥生成（KeyGen）：以公开参数为输入、输出公私钥对（$X \in \mathrm{PK}, x \in \mathrm{SK}$）的概率算法。

(3) NT 签名（NTSign）：一个概率算法，以 $<X_i, X_j, x_i, h_j, m>$（$X_i \neq X_j \in \mathrm{PK}$，$x_i \in \mathrm{SK}, m \in M, h_j \in F$）为输入，输出对 m 的签名 $\sigma = <s, h_i, h_j>$（$s \in S, h_i \in F$）。为了方便，称 σ 为 NT 签名（满足不可传递性的签名）。

(4) NT 签名验证（NTVerify）：以 $\bar{s} = <\sigma, X_i, X_j>$ 为输入，输出接收或拒绝。

(5) 并发签名验证（CSVerify）：以 $<k, \mathrm{kpos}, \bar{s}>$ 为输入（这里 k 是 Keystone，$\mathrm{kpos} \in \{1,2\}$）的算法。如果 kpos=1，检验是否满足 KCommit(k)=h_i，如果不是，拒绝并终止；如果 kpos=2，检验是否满足 KCommit(k)=h_j，如果不是，拒绝并终止。然后运行 NTVerify($\bar{s}$)，输出结果（拒绝或接受）。

下面给出一个实际示例[24]。

(1) 参数建立（Setup）：设安全参数为 l，p、q 是满足 $q \mid (p-1)$ 的大素数；$G = <g>$ 是 Z_p^* 的 q 阶子群；H_1, H_2：$\{0,1\}^* \to Z_q$ 是 2 个 Hash 函数；KCommit 定义为 H_1；公开参数 $<p, q, g, H_1, H_2>$。

(2) 密钥生成（KeyGen）：用户 1 和用户 2 的公、私钥分别为 $<x_i, X_i = g^{x_i} \bmod p>$（$i = 1, 2$）。

(3) NT 签名（NTSign）：不失一般性，无妨设输入 $<X_1, X_2, x_1, h_2, m>$。随机选取 $t \in Z_q$，计算

$$
\begin{aligned}
h &= H_2(X_1 \parallel X_2 \parallel m \parallel g^t X_2^{h_2}) \\
h_1 &= (h - h_2) \bmod q \\
s &= t - x_1 h_1 \bmod q
\end{aligned}
$$

输出签名 $\sigma = <s, h_1, h_2>$。

(4) NT 签名验证（NTVerify）：算法以 $<\sigma, X_1, X_2, m>$ 为输入，检验以下等式是否

成立：

$$(h_1+h_2)\bmod q = H_2(X_1 \parallel X_2 \parallel m \parallel g^s X_1^{h_1} X_2^{h_2} \bmod p)$$

(5) 并发签名验证(CSVerify)：设输入 Keystone k。检验 $H_1(k)=h_2$ 是否成立，如果成立再执行验证算法 NTVerify。

为方便起见，以下将该并发签名协议记为 CSig。

下面简要介绍一下并发签名协议的安全模型。

正确性定义是显然的，这里不予详述。

称并发签名协议满足不可传递性，如果满足：存在 PPT 算法 FakeNTSign，对于任意输入 $<X_1, X_2, x_2, m>$(用户 2 的私钥可以已知)，输出 NT 签名 σ 可以被 NTVerify 接受，并且和用户 1 产生的实际签名是计算不可区分的。

可以通过敌手 E 和挑战者 C 之间的一个 Game 来形式化定义不可伪造性。

Game 8.1

(1) 初始化：C 运行参数建立和密钥生成算法为用户分配参数和公、私钥。

(2) 敌手 E 可以向 C 做以下询问：

① KCommit 询问：E 可以要求 C 随机选择 Keystone k，返回公开承诺值 $f=\text{KCommit}(k)$；如果愿意自己选择 Keystone，则可以自己计算出 $f=\text{KCommit}(k)$。

② KReval 询问：E 可以要求 C 提供以前的 Kcommit 询问的输出所对应的 Keystone。

③ NTSign 询问：E 可以询问对应于 (X_i, X_j, h_j, m) 的 NT 签名 $\sigma=(s, h_i, h_j)=\text{NTSign}(X_i, X_j, h_j, m)$。

④ FakeNTSign 询问：E 可以要求获得对应 (X_i, X_j, m) 的 NT 签名，C 用 $\sigma=(s, h_i, h_j)=\text{FakeNTSign}(X_i, X_j, x_j, m)$ 作为回答。

⑤ Corrupt 询问：E 可以要求获得某用户的私钥。

(3) Game 输出：最后 E 输出对应于公钥 X_c、X_d，对某消息 m 的签名 $\sigma=(s, h_c, f)$，如果以下条件满足，就称 E 赢得 Game：$\text{NTVerify}(\sigma, X_c, X_d, m)$ 为真；以前没有任何形如 (X_c, X_d, f', m)(对任意的 $f' \in F$)的 NTSign 询问；没有形如 (X_c, X_d, m) 的 FakeNTSign 询问；没有针对 X_c 的 Corrupt 询问；下面两种情形之一成立：

① 没有针对 X_d 的 Corrupt 询问。

② 或者 f 以前是某 KCommit 询问的输出，或者 E 也输出一个满足 $f=\text{KCommit}(k)$ 的 Keystone k。

称并发签名是不可伪造的，如果任何 PPT 敌手赢得 Game8.1 的概率都是可忽略的。

仍然可以通过 Game 方法来定义公平性。

Game 8.2

(1) 初始化：同 Game 8.1。

(2) Oracle 询问：同 Game 8.1。

(3) 输出：最后敌手 E 输出一个 Keystone k 和$(\sigma=(s, h_c, f), X_c, X_d, m)$，签名能够通过 CSVerify 算法验证，如果以下情形之一成立，就称敌手赢得 Game。

① f 是以前某 KCommit 询问的输出，并且没有以 f 为输入的 KReval 询问。

② E 还输出一个$(\sigma'=(s', h_c', f), X_d, X_c, m')$，能够通过 NTVerify 验证，但不能通过 CSVerify 验证。

如果任何 PPT 敌手赢得 Game8.2 的概率都是可忽略的，称并发签名满足公平性。

需要指出的是，文献[26]中有关公平性的定义是存在问题的，因为从 Game 8.2 来看，情形(2)其实不可能发生：在($\sigma=(s,h_c,f)$，X_c，X_d，m)能够通过 CSVerify 验证的条件下，E 再输出一个能够通过 NTVerify 验证但不能通过 CSVerify 验证的签名($\sigma'=(s',h_c',f)$，X_c，X_d，m')是不可能的，因为二者共享 f=KCommit(k)。情形(2)的出发点是试图确保一旦 Keystone 揭示，收、发双方各自的签名都与其身份绑定，因此只需保证：对任意固定的 f，E 在至多只能掌握一方私钥(Corrupt 询问)的条件下无法同时伪造 2 个形如($\sigma=(s,h_c,f)$，X_c，X_d，m)、($\sigma'=(s',h_c',f)$，X_d，X_c，m')的合法签名。容易看出，在伪造 Game8.2 中已经包含了这种情况，因此，认为公平性 Game 8.2 的情形(2)并不必要。当然上述 Game 还应该确保 E 不能同时询问两个用户的私钥，显然文献[26]中的公平性 Game 忽视了这一点。

文献[26]中已证明以下定理。

定理 8.5 CSig 是安全的并发签名协议，即满足正确性、非传递性、不可伪造性和公平性。

8.13 强指定验证方的签名协议

"强指定验证方签名"(Strong Designated Verifier Signature，SDVS)是 Jakobsson 等人于 1996 年提出的一种签名[30]，与指定验证方签名(DVS)相比，SDVS 提供了更强的安全保证，除了具备 DVS 的所有性质之外，由于验证签名时必须要使用指定验证方的私钥(验证 DVS 签名并不需要验证方的私钥)，因此，SDVS 签名能够确保只有指定的验证方可以验证签名，这种性质称为强壮性(Strongness)。Saeednia 等人[31]于 2003 年给出了一个基于签密思想的 SDVS 协议，较深入地讨论了相关安全概念，但安全模型和安全性证明并不严格；Susilo 等人[32]于 2004 年设计了第一个基于身份的 SDVS 协议。另外，Ji-Seon 等人[33]于 2006 年指出可以利用消息恢复签名的设计思想构造高效的 SDVS。

8.13.1 2 个 SDVS 协议及其安全性分析

1. Saeednia 等人的 SDVS 协议[31]

设 p、q 是 2 个大素数，$q|(p-1)$；g 是乘法群 Z_p^* 的生成元；H：$\{0,1\}^* \to Z_p$ 是安全的 Hash 函数；m 是任意被签消息；用户 U 的私钥为 $x_U \in Z_q^*$，对应的公钥为 $y_U = g^{x_U} \bmod p$，密钥对记为(x_U，y_U)。

设用户 A 要产生 SDVS 签名并发送给用户 B。

签名的产生：A 随机选择 2 个数，$k \in Z_q$，$t \in Z_q^*$，计算

$$c = y_B^k \bmod p$$
$$r = H(m,c)$$
$$s = (kt^{-1} - rx_A) \bmod q$$

输出对消息 m 的签名(r，s，t)。

签名的验证：B 收到以上签名后，只需检验等式 $H(m,(g^s y_A^r)^{tx_B} \bmod p)=r$ 是否成立即可。

注意在验证签名时用到了接收方 B 的私钥，因此除了 B 之外任何一方应该无法验证签名。另外，即使 B 泄露其私钥 x_B，B 也仍然无法使任何第三方相信签名的合法性，因为 B 自己可以生成与 A 的真实签名计算不可区分的“模仿签名”：

B 可以随机选择 $s'\in Z_q$，$r'\in Z_q^*$，计算

$$c = g^{s'}y_A^{r'} \bmod p$$

$$r = H(m,c)$$

$$l = r'r^{-1} \bmod q$$

$$s = s'l^{-1} \bmod q$$

$$t = lx_B^{-1} \bmod q$$

输出模仿签名 (r,s,t)。

易于验证模仿签名和真实签名是计算不可区分的。

文献[25]中首次较详细地给出了有关安全概念，给出了部分安全性证明思路，但并未给出严格规范的完整证明，特别是没有证明协议满足强壮性(Strongness)。

2. Ji-Seon 等人的 SDVS 协议[33]

文献[33]利用消息恢复签名的设计思想，给出了一种效率较高的 SDVS 协议。

参数和密钥对选取与 Saeednia 等人的 SDVS 协议类似。

签名的产生：A 随机选择 2 个数，$k_2\in Z_q$，$k_1\in Z_q^*$，计算

$$c = g^{k_1} \bmod p$$

$$r = my_B^{k_2} \bmod p$$

$$s = k_1^{-1}(rx_A - k_2) \bmod q$$

输出对消息 m 的签名 (c,r,s)。

签名的验证：B 收到以上签名后，只需检验等式 $m(c^{-s}y_A^r)^{x_B} \bmod p = r$ 是否成立即可。

显然，在验证签名时同样用到了接收方 B 的私钥。另外，B 自己可以生成与 A 的真实签名计算不可区分的“模仿签名”：

B 可以随机选择 $t_2\in Z_q$，$t_1\in Z_q^*$，计算

$$c = y_A^{t_1^{-1}} \bmod p$$

$$r = mc^{x_B t_2} \bmod p$$

$$s = (t_1 r - t_2) \bmod q$$

输出模仿签名 (c,r,s)。

遗憾的是，上述 SDVS 协议存在如下缺陷：任何第三方都可以伪造合法SDVS 签名。

根据验证方程易于推导得到

$$m = rc^s y_A^{-rx_B} \bmod p$$

因此，只要观察到一个合法签名 (c_1,r_1,s_1)，就可得到 $m_1 = r_1 c_1^{s_1} y_A^{-r_1 x_B} \bmod p$，从而通过开方运算易于得到 $y_A^{x_B}$。此后，任意第三方通过随机选取 (c,r,s)，定义 $m = rc^s(y_A^{x_B})^{-r} \bmod p$，就得到了 m 的一个以 B 为指定验证方的合法签名。

尽管文献[33]中的协议存在安全缺陷，但所给出的利用具有消息恢复特性的数字签名构造高效 SDVS 协议的想法本身还是很有意义的，因为使用具有消息恢复性质的签名可以

提高实现速率、显著减少签名长度。下面将基于单向认证密钥交换协议和具有消息恢复特性的数字签名给出上述 SDVS 协议的一个改进。

8.13.2 SDVS 协议的安全模型

借鉴文献[26]对一般指定验证方签名(DVS)的定义方式给出了 SDVS 协议的严格形式化定义。强指定验证方签名(SDVS)协议定义为以下一组算法。

(1) 参数建立(Setup(1)):以安全参数 1 为输入的概率多项式时间(PPT)算法,返回系统参数 params,该参数包括以下空间的描述:公钥空间 PK、私钥空间 SK、消息空间 M 及签名空间 S。

(2) 密钥生成(KeyGen(params)):以系统参数 params 为输入、输出公钥 $X\in$ PK 及对应私钥 $x\in$ SK 的 PPT 算法。

(3) SDV 签名(SDVSign(X_S,X_V,x_S,m)):以签名方的公钥 $X_S\in$ PK 及对应私钥 $x_S\in$ SK、指定验证方的公钥 $X_V\neq X_S$、消息 $m\in M$ 为输入,以签名 $\sigma\in S$ 为输出的 PPT 算法。

(4) SDV 验证(SDVVerify(X_V,x_V,X_S,m,σ)):以指定验证方的公钥 $X_V\in$ PK 及对应私钥 $x_V\in$ SK、签名公钥 X_S、消息 $m\in M$、签名 $\sigma\in S$ 为输入,以接受(1)或拒绝(0)为输出的 PPT 算法。

与文献[25]不同的是,给出的 SDVS 定义采用了形式化方法,而且没有把“可模仿性”(即描述指定验证方如何模仿生成 SDVS 的算法)作为定义的一部分,而是将借鉴环签名的做法,通过把非传递性(Non-Transferability)作为 SDVS 安全性要求的一部分。

下面将给出 SDVS 的严格安全性定义。

如果满足:输入(X_S,X_V,x_S,m),运行 SDVSign 算法输出签名 σ,则以(X_V,X_S,x_V,m,σ)为输入的 SDVVerify 算法输出接受,称 SDVS 是正确的。

如果存在一个以(X_V,X_S,x_V,m)为输入的 PPT 算法 FakeSDVSign,输出签名 σ' 被 SDVVerify 接受且 σ' 的分布与 SDVSign 产生的“真实”签名 σ 是计算不可区分的(即使私钥 x_S、x_V 均已泄露),称 SDVS 是非传递的。

可以利用一个在挑战方 C 和敌手 E 之间运行的 Game 来形式化定义不可伪造性。

Game 8.3

(1) 初始化:C 运行 Setup 和 KeyGen 算法,产生系统参数和用户公、私钥对(X_i,x_i),这里 $i=1,2,\cdots,n$,n 表示用户数上界。E 只得到公开参数和所有用户的公钥。

(2) E 可以向 C 提出以下询问:

① SDVSign 询问:E 可以要求得到对应(X_S,X_V,m)的 SDVS 签名,C 运行 SDVSign 输出对应签名 σ 作为回答。

② FakeSDVSign 询问:E 可以要求得到对应(X_S,X_V,m)的一个伪造 SDVS 签名,C 运行以(X_V,X_S,x_V,m)为输入的 FakeSDVSign 算法,输出签名 σ' 作为回答。

③ Corrupt 询问:E 可以要求得到用户 i 的私钥 x_i。

④ SDVVerify 询问:这是 E 拥有的验证 Oracle,对于询问(X_S,X_V,m,σ),输出接受或拒绝。

(3) 输出:最后 E 输出签名(X_S^*,X_V^*,m^*,σ^*)。如果该签名被 SDVVerify 接受,且从未使用(X_S^*,X_V^*,m^*)向 Oracle SDVSign 或 FakeSDVSign 提出过询问,且从未针对

(X_S^*, X_V^*)提出过 Corrupt 询问，就称 E 赢得 Game。

如果任何 PPT 敌手 E 赢得 Game8.3 的概率是可忽略的（安全参数的可忽略函数），称 SDVS 是不可伪造的。

如果满足：除了指定验证方外，对于任何不知道验证方私钥的 PPT 敌手 E 而言，由任何第三方产生的“签名”σ'与真实签名 σ 是计算不可区分的，称 SDVS 是强壮的。

实际上，上述定义的 SDVS 的强壮性是指，区分真实签名的唯一途径是已知指定验证方的私钥，因此即使是签名方本人也不能区分真伪签名。

如果它满足正确性、非传递性、不可伪造性和强壮性，称一个 SDVS 协议是安全的。

8.13.3　基于环签名构造的 SDVS 协议

文献[26]中指出，任何一个 DVS 协议和 2 方环签名协议是等价的。但由于环签名协议的“公开可验证性”，因此并不能以平凡方式给出 SDVS 协议的构造方式。本节介绍一个基于 8.11 节的环签名构造的可证明安全的 SDVS 协议。

基于 RSS 构造 SDVS 协议的基本思想是：把判定性 Diffie-Hellman(DDH)问题“嵌入”到 RSS 的 2 方情形。记基于 RSS 构造的 SDVS 协议为 RSS-SDVS 协议。

RSS-SDVS 协议可由以下一组算法定义。

(1) Setup：与 RSS 类似，即给定某安全参数 l，设 p、q 是 2 个大素数，满足$q|(p-1)$；设 G 是 Z_p^* 的 q 阶乘法群，g 是 G 的生成元；设 H、H'是 2 个从$\{0,1\}^*$映到 Z_q 的安全 Hash 函数，公开参数 params=(p,q,g,H,H')。注意这里需要引入2 个独立的 Hash 函数。

(2) KeyGen：该算法以 params 为输入，输出用户私钥 $x\in_R Z_q^*$ 和公钥$X=g^x \bmod p$。

(3) SDVSign：输入消息 m、签名方 A 和指定验证方 B 的公钥(X_A, X_B)以及签名方私钥 x_A，A 随机选择 $h_2, k\in_R Z_q^*$，计算

$$z = X_B^{h_2} g^k \bmod p$$
$$c = H'(m, X_A, X_B, g^k, X_B^k)$$
$$h = H(X_A, X_B, c, z)$$
$$h_1 = (h - h_2) \bmod q$$
$$s = (k - x_A h_1) \bmod q$$

输出签名 $\sigma=(s, h_1, h_2)$。

(4) SDVVerify：以指定验证方 B 收到的消息 m、公钥(X_A, X_B)以及验证方私钥 x_B、签名 $\sigma=(s, h_1, h_2)$为输入，计算

$$g^k = g^s X_A^{h_1} \bmod p$$
$$c' = H'(m, y_A, y_B, g^k, g^{kx_B})$$
$$h = H(y_A, y_B, c', z)$$
$$h_1 + h_2 = h \text{ 是否成立？}$$

如果等式成立，输出接受；否则拒绝。

显然，上述 SDVS 协议本质上是从 2 方环签名即 2 方-RSS 演变而来，RSS-SDVS 协议的不可伪造性可以归约为 RSS 签名的不可伪造性。

下面来给出 RSS-SDVS 协议的安全性证明。

RSS-SDVS 协议的正确性证明是显然的。

定理 8.6　RSS-SDVS 协议是非传递的。

证明　设 $\sigma=(s,h_1,h_2)$ 是签名方 S 以 (X_S,X_V,x_S,m) 为输入运行 SDVSign 算法生成的对任意消息 m 的一个"真实"SDVS 签名，这里 (X_S,x_S)，(X_V,x_V) 分别是签名方 S 和指定验证方 V 的公、私钥对。下面将定义一个算法 FakeSDVSign，它是以 (X_V,X_S,x_V,m) 为输入的 PPT 算法，输出签名 $\sigma'=(s',h_1',h_2')$：随机选择 $h_1',k\in Z_q$，计算

$$z=X_S^{h_1'}g^k \bmod p$$

$$c=H'(m,X_S,X_V,g^k,X_V^k)$$

$$h=H(X_S,X_V,c,z)$$

$$h_2'=(h-h_1')\bmod q$$

$$s=(k-x_Vh_2')\bmod q$$

显然签名可以被 SDVVerify 算法接受。下面证明 $\sigma'=(s',h_1',h_2')$ 和 $\sigma=(s,h_1,h_2)$ 是计算不可区分的。假设 H、H' 都是随机预言[34]，易于看出，σ' 和 σ 的每一个签名分量都在 Z_q 上均匀分布，但 3 个签名分量并不独立：以 $\sigma'=(s',h_1',h_2')$ 为例，它们满足以下等式

$$h_1'+h_2'=H(X_S,X_V,c,g^{s'}X_S^{h_1'}X_V^{h_2'})$$

$$c'=H'(m,X_S,X_V,g^k,X_V^k)$$

真实签名 $\sigma=(s,h_1,h_2)$ 有完全相同的等式关系成立

$$h_1+h_2=H(X_S,X_V,c,g^sX_S^{h_1}X_V^{h_2})$$

$$c=H'(m,X_S,X_V,g^k,X_V^k)$$

注意到两个等式的对应输入的分布完全相同，均为均匀分布，因此 $\sigma'=(s',h_1',h_2')$ 和 $\sigma=(s,h_1,h_2)$ 的分布完全相同，当然也是计算不可区分的。

要证明 SDVS 协议的不可伪造性，需要用到 Gap 假设[35]。Gap 问题就是这样一个问题，在判定性 Oracle(如 DDH Oracle)的帮助下，解决某计算问题(如 CDH 问题)。目前 Gap 问题已经在可证明安全研究领域获得了很多成功应用[34]，也逐渐被学术界所接纳。这里先回顾一下有关的几个问题。

(1) **DLP**：已知 $g^a\in G(a\in_R Z_q)$，求解离散对数值 a。

(2) **CDH 问题**：已知 $g^a,g^b\in G(a,b\in_R Z_q)$，求解 $g^c=g^{ab} \bmod p$。

(3) **DDH 问题**：已知 $g^a,g^b,g^c\in G(a,b\in_R Z_q)$，判别 $c=ab$ 是否成立。

(4) **GDH 问题**：已知 $g^a,g^b\in G(a,b\in_R Z_q)$，同时已知一个可以解决 DDH 问题的 Oracle，求解 $g^c=g^{ab} \bmod p$。

如果在群 G 上 GDH 问题是难解的，则称 G 是一个 GDH 群。

如果求解以上问题的任何 PPT 算法的成功概率是可忽略的，就称求解以上问题是困难的。以上假设中显然 DLP 是最弱的，因此任何签名如果在 DLP 假设成立的条件下不可伪造，则在其他 3 个假设中任何一个成立的意义下必然也是不可伪造的。

定理 8.7　如果在 GDH 群上 Schnorr 签名是不可伪造的，即能抵抗适应性选择消息伪造攻击，则 RSS-SDVS 协议也是不可伪造的。

证明　根据已有结论"如果 Schnorr 签名是不可伪造的，则环签名 RSS 也是不可伪造的"[26,27]，事实上只需证明：如果 2 方环签名即环签名 2-RSS 是不可伪造的，则 SDVS 协议

也是不可伪造的。采用反证法。假设 H、H'都是随机预言[34]，假设存在敌手 E 在至多提出 q_H 次随机预言询问和至多 q_S 次 SDVSign 和 FakeSDVSign 询问后，以不可忽略概率 η 赢得不可伪造性 Game。现在就来证明存在以 E 为子程序的算法 Sim，以不可忽略概率伪造一个合法的 2-RSS 签名。

设 Sim 试图伪造的 2-RSS 签名的公钥列表为($X=g^x \bmod p, X'=g^{x'} \bmod p$)，它有一个 RSS 签名 Oracle C，C 可以回答 Sim 对任何消息 m 的 2-RSS 签名 $\sigma=(s,h_1,h_2)$。

Sim 开始和 E 运行 Game。首先进行初始化，即取公开参数(p,q,g)，设定用户总数为安全参数的多项式函数 $n(l)$；接着运行 KeyGen，为每一个用户生成公、私钥(X_i,x_i)，但对于一对随机选择的用户 A、B 例外：A 的公钥选为 $X_A=X$，B 的公钥选为 $X_B=X'$。Sim 向 E 提供所有用户的公钥。现在 Sim 向 E 模仿挑战方的行为，即对 E 可能提出的 Oracle 询问进行模仿回答。

随机预言 H 或 H' 询问：注意 Sim 维持着两个随机预言回答列表 L_H 和 $L_{H'}$，开始为空。对 H'询问给予随机回答，注意如果(X_S,X_V)=(X_A,X_B)(或顺序相反)，且随机预言询问形如(m,X_S,X_V,g^a,g^c)，Sim 需要询问 DDH Oracle g^c 是否为(X_V,g^a)的 Diffie-Hellman 值，如果不是，随机回答，如果是，查看 $L_{H'}$，如果存在标记为(($m,X_S,X_V,g^sX_V^{h_2}$,?),c)的值，以 c 作为回答，否则随机回答。对 H 询问则转交给 C 来回答。注意：只是在回答 H'询问时用到了 GDH 假设。

SDVSign 询问：E 可以询问对应任意(X_S,X_V,m)的 SDVS 签名，如果 $X_S \neq X_A=X$ 且 $X_V \neq X_B=X'$(或 $X_V \neq X_A=X$ 且 $X_S \neq X_B=X'$)，因为 2 个对应私钥均已知，Sim 不需要模仿，直接运行 2 方情形的 RingSign 算法即可给出回答；否则如果 $X_S=X_A$ 且 $X_V \neq X_B=X'$，首先随机选择 $s,h_1,h_2 \in Z_q$，计算 $K=g^sX_s^{h_1}$，$c=H'(m,X_S,X_V,K,K^{x_V})$定义 $h=h_1+h_2=H(X_S,X_V,c,KX_V^{h_2})$，输出 $\sigma=(s,h_1,h_2)$作为回答；如果 $X_S=X'$且 $X_V \neq X_A=X$，回答方式类似；如果 $X_S=X_A$ 且 $X_V=X_B$(或顺序相反)，首先随机选择 $c \in Z_q$，向 C 询问 c 的 2-RSS 签名，设得到的回答是 $\sigma=(s,h_1,h_2)$，定义 $c=H'(m,X_S,X_V,g^sX_V^{h_2},?)$，并把($m,X_S,X_V,g^sX_V^{h_2}$,?)和对应 c 值填入表 $L_{H'}$ 中。如果 E 曾经向 Oracle H'询问过($m,X_S,X_V,g^sX_V^{h_2}$,?)，宣布失败，否则把对应输出 $\sigma=(s,h_1,h_2)$作为回答。由于 $g^sX_V^{h_2}$ 是均匀分布的，因此模仿失败的概率显然可以忽略。

FakeSDVSign 询问：E 可以询问对应任意(X_S,X_V,m)的伪造 SDVS 签名，回答方法与 SDVSign 基本是一样的。

Corrupt 询问：E 可以询问对应任何公钥 X_i 的私钥 x_i。如果 $X_i=X_A=X$ 或 $X_i=X_B=X'$，Sim 放弃模仿；否则 Sim 输出私钥 x_i 作为回答。

Output：在 Game 中止时，E 输出签名(X_S^*,X_V^*,m^*,σ^*)。如果该签名被 SDVVerify 接受且以下条件满足：$X_S^*=X_A=X$，$X_V^*=X_B=X'$(或顺序恰好相反)；从未使用(X_S^*,X_V^*,m^*)向 Oracle SDVSign 或 FakeSDVSign 提出过询问；从未针对(X_S^*,X_V^*)提出过 Corrupt 询问，就称 E 赢得 Game。

因为 A、B 是随机选择的，因此事件"$X_S^*=X_A=X$，$X_V^*=X_B=X'$(或顺序恰好相反)"发生的概率至少为 $2/n^2$，显然 X_A、X_B 没有被提出过 Corrupt 询问，这时 Sim 必然没有放弃，设 $\sigma^*=(s^*,h_1^*,h_2^*)$，因为 H'是随机预言，除了可忽略情形，E 必然向 Oracle H'询问过

$(m^*, X_S, X_V, g^{s^*} X_S^{h_1^*}, (g^{s^*} X_S^{h_1^*})^{x_V})$，设给出的回答为 $c^* = H'(m^*, X_S, X_V, g^{s^*} X_S^{h_1^*}, (g^{s^*} X_S^{h_1^*})^{x_V})$，那么 Sim 输出 $\sigma^* = (s^*, h_1^*, h_2^*)$作为对消息 c^* 的 2-RSS 环签名，显然至多除了可忽略概率 Sim 没有向 C 询问过 c^*，因此 Sim 至少以不可忽略概率成功伪造了一个 2-RSS 签名。这与环签名 RSS 的不可伪造性矛盾。

最后考虑 SDVS 的强壮性(Strongness)。显然，要验证 SDVS 签名的合法性，必须使用验证方的私钥来计算 Diffie-Hellman 值 c(否则由于 CDH 假设，在随机预言模型中敌手正确猜测 H'值 c 的概率必然是可忽略的)，才能达到验证签名合法性的目的。因此，在未知指定验证方私钥的条件下，可以看出如果 DDH 假设满足，则真实签名和相同结构的随机数组是计算不可区分的。

8.13.4 基于可否认的单向认证密钥交换协议构造的 SDVS 协议

我们将以单向认证密钥交换协议(One-Pass Authentication Key Exchange protocol，简记为 OP-AKE 协议)作为基本工具，给出另外一种 SDVS 协议的一般构造方法。因为很多 AKE 协议都有仅需 1 次数据流动的变形协议，因此应用意义具有一般性。

首先讨论可否认的 OP-AKE 协议的安全模型。

1. 可否认的 OP-AKE 协议及其安全模型

OP-AKE 协议最早是由 Nyberg 和 Rueppel[36] 于 1993 年提出的，随后又于1994 年[37] 对协议作了进一步的推广和改进。OP-AKE 协议的基本内含是：任何两个通信方 A、B，发送方 A 只需向接收方 B 传递一次消息(1 次交互)，A、B 得以建立共享会话密钥(准确地说，就是只要双方都是诚实的，那么只有 A、B 才可能获得唯一的共享会话密钥 sk)。也就是说，B 知道消息源于 A 而且意定的接收方就是自己。因此 OP-AKE 协议的安全性要求和一般 AKE 协议是类似的。当然不能期望OP-AKE协议的安全强度和一般 AKE 协议完全等价，如在 OP-AKE 协议中，接收方 B 收到的消息可能是敌手的重放，在一次数据流动情形，这种重放几乎是不可避免的，但在实际应用中可以采取多种措施来弥补这一点，如采用时间戳机制或建立递增序列号机制等。另外，OP-AKE 协议也不能达到完善前向安全性，因为已知 B 的私钥足以推出共享会话密钥。但无论如何，由于只需要一次数据交互，OP-AKE 协议仍具有重要的应用意义。

可否认的 AKE 协议的设计思想最早是在由文献[38]中提出的。基本想法是：设计密钥交换协议时，既要考虑满足一般意义下的 AKE 协议的安全性，同时还要满足可否认性，即消息的发送方或所有者在必要时可以否认曾经发送过该消息(任何一方不具有向第三方证实曾经发生过会话的能力)。在很多需要保护用户隐私的情况下，可否认的 AKE 协议具有重要的应用价值。注意一般 AKE 协议未必自然满足可否认性，因为为了抵抗诸如密钥替换之类的攻击，往往需要在协议会话中加入用户身份信息，而且广泛使用数字签名、消息认证码等工具，这会使得协议会话记录与用户身份绑定。

下面将以 mBJM 模型为基础建立可否认的 OP-AKE 协议的安全模型。有关 mBJM 模型的详细知识可参见文献[26]，实际上是 AKE 协议在 PKI 环境中的BR 安全模型[39] 的推广应用。

首先考虑 OP-AKE 协议作为一般 AKE 协议的安全性。在 mBJM 模型中，每个用户

U 具有对应的公私钥(PK_U,SK_U),通常使用 Oracle $\prod_U^i$ 形式化表示用户 U 的第 i 个通信实例,这些 Oracle 对各类输入的消息(询问)依据协议规则给出相应的输出(回答)。任何 $\prod_U^i$ 只可能处于 3 种状态之一:未决(Undecided)状态、接受状态和拒绝状态。一旦处于接受状态,$\prod_U^i$ 应该具有:角色 $role_U^i \in$ {发起方,应答方};对应伙伴(Partner)的 ID,记为 pid_U^i(即意定的通信方);会话 ID sid_U^i 以及会话密钥 sk_U^i。

敌手 E 被形式化为一个 PPT 算法,E 完全控制信道,并通过提出 Oracle 询问的方法与 Oracles 交互。

协议的安全性形式化为以下的在挑战者 C 和敌手 E 之间的一个"Game"。

Game 8.4

(1) C 作为模仿方运行参数建立和密钥生成算法为用户分配参数和公私钥(E 不知道私钥)。

(2) E 可以提出多项式数量级的 Oracle 询问,由 C 给出回答。E 可以提出以下询问:

① Send ($\prod_U^i$,M):E 向发起方 Oracle $\prod_U^i$ 发送消息 M,C 根据协议给出模拟回答,注意在 OP-AKE 协议中 Send 询问只能针对发起方。

② Reveal ($\prod_U^i$):E 要求获得 $\prod_U^i$ 持有的会话密钥 sk_U^i。

③ Corrupt(U):E 要求获得 U 的私钥。

如果 E 曾经提出 Reveal 询问,称 E 已经揭示了 Oracle $\prod_U^i$;如果 E 曾经提出 Corrupt 询问,称 E 已经收买了相应的用户;如果 $\prod_U^i$ 的伙伴没有被揭示且 pid_U^i 没有被收买,称该 Oracle 是新鲜的(Fresh)。

④ Test ($\prod_{U^*}^i$):在某一时刻,E 可以对某新鲜 Oracle $\prod_{U^*}^i$ 提出 Test 询问,C 随机选择比特 b。如果 b 为 1,C 输出 $sk_{U^*}^i$,否则输出随机数。此后 E 可以继续提出前面的 Oracle 询问,但禁止揭示 $\prod_{U^*}^i$ 及其伙伴 Oracle,也不能收买 $pid_{U^*}^i$。

(3) 敌手 E 输出对比特 b 的猜测值(0 或 1)。

如果 E 对比特 b 的猜测正确,就称敌手赢得了 mBJM Game。

如果协议满足强相伴性,且任何 PPT 敌手赢得 mBJM Game 的概率优势是可忽略的,就称 OP-AKE 协议满足 mBJM-AKE 安全性。这里 AKE 协议的强相伴性是指:对于任何两个非意定的通信方而言,任何 PPT 敌手不可能以不可忽略概率使得二者均处于接受状态并持有相同的会话密钥。有关 mBJM 模型的细节可参见文献[26]。

下面考虑可否认性。

与文献[38]中的方式类似,采用模仿观点给出 OP-AKE 协议的可否认性定义。基本想法是:协议应答方对协议的观察(View)可以被任何不知道发起方私钥的模仿器 Simulator

模仿,而且除了可以模仿发送方的发送消息记录,还可以输出模仿会话密钥 K。模仿会话密钥 K 是与普通的可否认认证协议的主要区别,这样做是考虑到 K 很可能作为其他协议的输入,可否认认证协议仅需简单输出 1b,即接受或拒绝。由于 OP-AKE 协议属于单向交互协议,协议应答方并不发送任何消息,因此只需考虑发起方具有否认能力,即应答方无法向第三方证明:意定发起方曾经成功地向他发送了协议交互信息并达成某会话密钥 K。

称 OP-AKE 协议是可否认的,如果对于任意如上敌手 E,存在某模仿器 SIM_E,其输入与 E 完全一样,输出的模仿观察(Simulated View)和敌手 E 从协议执行得到的真实观察是计算不可区分的。

设两方 OP-AKE 协议的合法通信双方是 A、B,各自的公、私钥对为(X_A,x_A),(X_B,x_B),规定 A 是发起方,B 是应答方,协议输出会话密钥 K 或"error"(如果 B 的验证失败)。可否认的 OP-AKE 协议主要考虑应答方 B 不诚实的情况,安全目标是阻止 B 向第三方证明他与 A 达成了共享会话密钥。上述定义说明,不但 A 的发送消息是可以否认的,而且相关的会话密钥 K 也是可否认的。考虑敌手 E,输入为某意定数目的公钥集合 $PK=(pk_1,\cdots,pk_l)$,此外还有某些辅助输入。E 可以启动任意数目的用户间会话,这些会话是并发的,而且可以由 E 编排次序,但不考虑重放的情形。E 的观察包括随机输入、发起方发送的消息记录、E 参与的所有协议会话最终计算输出的会话密钥 K。

注意:*在该模型中,E 的输入还可以包括接收方的私钥,这表示即使不诚实的用户揭示自己的私钥也无法向第三方证明协议会话曾经发生过。*

2. 基于可否认的 OP-AKE 协议构造 SDVS 协议

这里给出基于可否认的 OP-AKE 协议构造 SDVS 协议的一般方法。

定理 8.8 安全的 SDVS 协议可以基于安全的可否认的 OP-AKE 协议构造。

证明 不失一般性,以 2 方情形为例给出证明,多方情形的证明是类似的。首先指出如何由 OP-AKE 协议构造 SDVS,然后讨论 OP-AKE 协议的安全性可以确保这样构造的 SDVS 协议也是安全的。

假设签名方为 S,指定验证方为 V,对应公、私钥分别为(X_S,x_S),(X_V,x_V)。

SDVS 协议的 Setup、KeyGen 算法与 OP-AKE 协议相同。

$\mathrm{SDVSign}(X_S,X_V,x_S,m)$:$S$ 作为发起方和应答方 V 执行 OP-AKE 协议,设 S 发送的消息为 T_S,S 计算出会话密钥 K,设 H 是某安全 Hash 函数,输出 $\sigma=(T_S,C=H(S,V,m,K))$ 作为对消息 m 的 SDVS 签名。

$\mathrm{SDVVerify}(X_S,X_V,x_V,m,\sigma)$:$V$ 根据协议规定,计算出会话密钥 K,以及 $C'=H(S,V,m,K)$,比较 $C'=C$ 是否成立。

易于看出,如果 OP-AKE 协议满足正确性,SDVS 协议必然也满足正确性。

其次,如果 OP-AKE 协议是可否认的,则存在模仿算法 SIM,除公开输入外,还包括 V 的私钥作为输入,在未知发送方私钥 x_S 的条件下输出模仿观察,包括 T、K,因此易于计算得到 $C=H(S,V,m,K)$,显然这样计算出的签名会被 SDVVerify 接受。因此非传递性满足。

下面考虑不可伪造性。不可伪造性 Game 和 OP-AKE 的安全性 Game 是基本类似的,回答 Oracle 询问的方法如下:

SDVSign 询问:当询问 m 的签名时,对应了 OP-AKE 安全性 Game 中的 Send 询问和

Kreveal 询问，因此可以自然给出 SDV 签名作为回答，注意该询问是针对 S 的。

FakeSDVSign 询问：和 SDVSign 询问的回答方式类似，但 Kreveal 询问是针对应答方 V 的。

对随机预言 H 的回答方式是随机的。总之在 SDVS 不可伪造性 Game 中的全部 Oracle 询问均可由 OP-AKE 协议安全性 Game 中的对应 Oracle 做出回答。最后，假设敌手未经询问 Oracle SDVSign 和 FakeSDVSign，输出某新消息 m^* 的签名 σ^*，因为是随机预言，敌手必定向 H 询问过(S,V,m^*,K^*)，显然如果 OP-AKE 是安全的，那么敌手成功伪造上述签名的概率必然也是可忽略的。

最后考虑强壮性(Strongness)。可以考虑这样的两阶段 Strongness Game。Setup、KeyGen 同上，在第一阶段，敌手通过询问 Oracle SDVSign 和 FakeSDVSign 等(回答方式同上)，敌手选择某消息 m 作为测试消息，需要挑战方以 1/2 概率给出真实签名 σ 或伪造签名 σ'，敌手输出判断 1 或 0。如果敌手的正确判断概率是不可忽略的，就称敌手成功，从而不满足强壮性，反之称满足强壮性。显然，这与 OP-AKE 协议的安全性 Game 的 Test 询问是对应的，即以概率 1/2 输出真实会话密钥或随机数，然后计算 $C=H(S,V,m,K)$。根据归约论断已知，如果真实 K 与随机数 R 是不可区分的，那么真实签名 σ 或伪造签名 σ' 必然也是不可区分的。

可以看出，基于可否认 OP-AKE 协议构造 SDVS 协议的方法是一般的，突出优点之一是，强壮性易于得到证明。

3. 2 个基于 OP-AKE 协议的 SDVS 协议实例

上面曾经指出，SDVS 协议的构造可以归结为可否认的 OP-AKE 协议的构造。目前有许多 OP-AKE 协议可供选择。首先利用文献[40]中提出的 HMQV 密钥交换协议的单向变形(记为 OP-HMQV 协议)，具体实现一个 SDVS 协议；其次，基于 Nyberg 和 Rueppel 等人的消息恢复签名机制[37]及对应 OP-AKE 修改协议，给出基于该协议的另一个 SDVS 实例。

基于 OP-HMQV 协议[37]的 SDVS 协议：

(1) Setup：给定某安全参数 l，设 p、q 是两个大素数，满足 $q\mid(p-1)$；设 G 是 Z_p^* 的 q 阶乘法群，g 是 G 的生成元；设 H、H_1、H_2 是 3 个从$\{0,1\}^*$映到 Z_q 的安全 Hash 函数，公开参数 params$=(p,q,g,H,H_1,H_2)$。

(2) keyGen：该算法以 params 为输入，输出用户私钥 $x\in_R Z_q^*$，公钥$X=g^x \bmod p$。

(3) SDVSign：输入消息 m、签名方 S 和指定验证方 V 的公钥(X_S,X_V)以及签名方私钥 x_S，S 随机选择 $x\in_R Z_q$，计算

$$\begin{aligned}X &= g^x \bmod p,\\ d &= H_1(X,X_S,X_V),\\ \sigma_S &= X_V^{x+dx_S},\quad K = H_2(\sigma_S),\\ C &= H(X_S,X_V,m,K)\end{aligned}$$

输出 SDVS 签名 $\sigma=(X,C)$。

(4) SDVVerify：输入消息 m 及签名 $\sigma=(X,C)$、签名方 S 和指定验证方 V 的公钥(X_S,X_V)及验证方私钥 x_V，V 首先检验是否 $X\neq 0$，如果是，计算 $d=H_1(X,X_S,X_V)$，$K=$

$H_2((XX_S^d)^{x_V})$，并检验验证方程 $C=H(X_S,X_V,m,K)$ 是否成立。

根据文献[37]及定理8.8，上述 SDVS 协议的正确性、不可伪造性和强壮性不难证明，因此下面只需给出非传递性证明。

定理 8.9 基于 OP-HMQV 协议的 SDVS 协议是非传递的。

证明 设 $\sigma=(s,h_1,h_2)$ 是签名方 S 以 (X_S,X_V,x_s,m) 为输入运行 SDVSign 算法生成的对任意消息 m 的一个“真实”SDVS 签名，这里 (X_S,x_S)，(X_V,x_V) 分别是签名方 S 和指定验证方 V 的公、私钥对。下面定义以 (X_V,X_S,x_V,m) 为输入的算法 FakeSDVSign，输出签名 $\sigma'=(s',h_1',h_2')$：随机选择 $x\in Z_q$，计算

$$X = g^x \bmod p,$$
$$d = H_1(X,X_S,X_V),$$
$$\sigma_S = (XX_S^d)^{x_V},\quad K = H_2(\sigma_S)$$
$$C = H(X_S,X_V,m,K)$$

显然，签名可以被 SDVVerify 算法接受。易于证明 σ' 和 σ 的分布完全相同，均为均匀分布，当然也是计算不可区分的。

cNR-SDVS 协议：

前面已经指出，基于消息恢复签名思想构造 SDVS 的基本思想是很有意义的，下面基于 Nyberg 和 Rueppel 的消息恢复签名思想，结合对其 OP-AKE 协议变形作适当修改，设计一个 SDVS 协议，无妨记为 cNR-SDVS 协议，仍然通过以下几个算法来定义。

(1) Setup 与 keyGen：设 p、q 是两个大素数，满足 $q|(p-1)$；设 G 是 Z_p^* 的 q 阶乘法群，g 是 G 的生成元；设 H、H' 是两个从 $\{0,1\}^*$ 映到 Z_q 的安全 Hash 函数，公开参数 params$=(p,q,g,H,H')$。用户私钥 $x\in_R Z_q^*$，公钥 $X=g^x \bmod p$。

(2) SDVSign：设 SDVS 签名方为 S，指定验证方为 V，被签消息为 m，S 执行以下过程。

① 随机选择 $K,k\in Z_q$。

② 计算 $r=X_B^K g^{-k} \bmod p$，约化 $r'=r \bmod q$。

③ 计算 $s=(k-r'x_S) \bmod q$。

④ 计算 $\text{sk}=H(X_S,X_V,r,g^K) \bmod p$。

⑤ 计算 $c=H'(X_S,X_V,m,\text{sk})$。

并输出 SDVS 签名 (c,r,s)。

(3) SDVVerify：指定验证方 V 执行：

① 恢复 $X_B^K \bmod p$：$X_B^K=g^x X_S^{r'} r \bmod p$。

② 计算 $g^K=(X_V^K)^{x_V^{-1}}$，$\text{sk}=H(X_S,X_V,r,g^K \bmod p)$。

③ 验证签名方程 $c=H'(X_S,X_V,m,\text{sk})$ 是否成立。

易于看出，cNR-SDVS 协议和文献[33]中的签名长度相同，计算复杂性相当，除了 cNR-SDVS 协议引入了两个独立的 Hash 函数(Hash 函数的处理速度通常是很快的)，特别是二者的不可伪造性质均依据文献[37]中提出的具有消息恢复性质签名协议即 Nyberg-Rueppel 签名协议，基于消息恢复签名设计 SDVS 的突出优点之一就是可以缩短签名长度。但前面已经分析指出，文献[33]中的 SDVS 不能抵抗伪造攻击。原因在于，给出 Nyberg-Rueppel 签名的存在性伪造是容易的。cNR-SDVS 协议没有对任意消息

m 直接签名,而是通过对特定格式消息 X_B^K 签名,然后基于 CDH 问题的变形问题(即已知 $g^{ab}, g^b \in G(a, b \in_R Z_q)$,求解 $g^a \bmod p$。显然和 CDH 问题等价),并通过随机预言 H 的随机化处理,得到对应 OP-AKE 协议的会话密钥 sk,然后作 HMAC 确认。迄今为止,对这类签名的分析结果表明,目前对上述特殊格式的消息 X_B^K 而言,很难给出已知指数 K 的 Nyberg-Rueppel 签名存在性伪造。因此基于 CDH 假设,cNR-SDVS 协议的抗伪造攻击性质是有保证的,此外注意,对会话密钥的计算引入了 Hash 函数处理,而且引入了其他随机因素,这主要是为了确保 SDVS 协议所依据的 OP-AKE 协议的协议会话满足强相伴性[26]。

根据以上分析及定理 8.8,如果在以上意义下 Nyberg-Rueppel 签名是不可伪造的,那么易于证明 SDVS 协议满足正确性、不可伪造性和强壮性,由于篇幅所限,这里就不再赘述,基本方法可以参考文献[26]中的模块化证明观点。下面考虑非传递性。

定理 8.10 cNR-SDVS 协议满足非传递性。

证明 设 $\sigma=(c, r, s)$ 是签名方 S 以 (X_S, X_V, x_s, m) 为输入运行 SDVSign 算法生成的对任意消息 m 的一个"真实"SDVS 签名,这里 (X_S, x_S),(X_V, x_V) 分别是签名方 S 和指定验证方 V 的公、私钥对。下面定义以 (X_V, X_S, x_V, m) 为输入的算法 FakeSDVSign,输出签名 $\sigma'=(c', r', s')$:随机选择 S 执行以下过程。

(1) 随机选择 $r, s \in Z_q$,约化 $r'=r \bmod q$。

(2) 计算 $g^k = X_S^{r'} g^s \bmod p$。

(3) 计算 $X_V^{K'} = r g^k \bmod p, g^{K'} = (X_V^{K'})^{x_V^{-1}}$。

(4) 计算 $sk' = H(X_S, X_V, r, g^{K'}) \bmod p$。

(5) 计算 $c' = H'(X_S, X_V, m, sk')$。

并输出签名 $\sigma'=(c', r', s')$。

显然,签名可以被 SDVVerify 算法接受,易于证明 σ' 和 σ 的分布完全相同,均为均匀分布,当然也是计算不可区分的。

8.14 小结

我们将数字签名协议分为两类:一类是普通数字签名协议,只提供签名生成和签名验证两个基本功能,而且这两个功能可分别由单个签名者和单个验证者独立完成,该类数字签名协议通常称为数字签名算法,如第 1 章中介绍的 RSA 数字签名算法、DSA;另一类是特殊数字签名协议,该类数字签名协议除了提供签名生成和签名验证两个基本功能外,还具有其他特定的辅助功能或需要联合签名或验证,如本章介绍的不可否认的数字签名协议、Fail-Stop 数字签名协议、群数字签名协议和盲签名协议等。

目前已提出了大量各种各样的数字签名协议,本章也介绍了一些典型的和最新的数字签名协议[41]。关于数字签名协议的综述论文可参阅文献[42]～[44],当然这些文献相对比较早,还不能覆盖最新研究成果。另外,由于篇幅的限制,只介绍了作者在这一领域的部分研究成果,感兴趣的读者还可参阅文献[45]～[50]。

参考文献

[1] Simmons G J. The prisoner's problem and the subliminal channel, Advances in Cryptology-Crypto'83, Plenum Press, 1984, 51～67.

[2] Simmons G J. The subliminal channel and digital signatures, Advances in Cryptology-Eurocrypto'84, Springer-verlag, 1985, 51～67

[3] Simmons G J. Subliminal communication is easy using the DSA, Advances in Cryptology-Eurocrypto'93, Springer-verlag, 1994, 218～232.

[4] Simmons G J. A secure subliminal channel(?), Advances in Cryptology-Crypto'85, Springer-verlag, 1986, 33～41.

[5] Desmedt Y, Goutier C, Bengio S. Special uses and abuses of the Fiat-Shamir passport protocol, Advances in Cryptology-Crypto'87, Springer-verlag, 1988, 21～29.

[6] Chaum D, Van Antwerpen H. Undeniable Signatures, Advances in Cryptology—Crypto'89, Springer-Verlag, 1990, 212～216.

[7] Van Heyst E, Pedersen T P. How to make efficient fail—stop signatures, Advances in Cryptology—Eurocrypt'92, Springer-Verlag, 1993, 366～377.

[8] Chaum D, Van Heijst E. Group Signatures, Advances in Cryptology—Eurocrypt'91, Springer-Verlag, 1991, 257～265.

[9] Chaum D. Blind signatures for untraceable payments, Advances in Cryptology-CRYPTO'82, Plenum Press, 1983, 199～203.

[10] Desmedt Y, Frankel Y. Threshold cryptosystems. In: Brassard G, ed. Proceedings of the Crypto'89. LNCS 435, Berlin: Springer-Verlag, 1990. 307～315.

[11] Desmedt Y, Frankel Y. Shared generation of authenticators and signatures. In: Desmedt Y, Frankel Y, eds. Advances in Cryptology-CRYPTO' 91, Lecture Notes In Computer Science, Berlin: Springer-Verlag, 1992, 457～469.

[12] De Santis A, Desmedt Y, Frankel Y. How to share a function securely. In Proceedings of the 26th ACM Symposium on the Theory of Computing, Santa Fe, 1994, 522～533.

[13] Gennaro R, Jarecki S, Krawczyk H. Robust Threshold DSS Signatures, In Ueli Maurer, editor. Advances in Cryptology—Eurocrypt 96, Lecture Notes in Computer Science. No. 1070. Springer Verlag, 1996, 354～371.

[14] 石怡，冯登国. 一类新型(t_j, t, n)-门限群签名方案的设计与分析. 见：王鄂芳，杨伟成，编. 密码学进展——ChinaCrypto 2000. 北京：科学出版社，2000. 156～159.

[15] 陈伟东，冯登国. 一类存在特权集的门限群签名方案. 软件学报. 2005. 16(7)：1289～1295. 又见 Chen Wei-dong, Feng Deng-guo. A Group of threshold group-signature schemes with privilege subsets, International Workshop on Progress on Cryptography: 25 years of Cryptography in China, KLUWER ACADEMIC PUBLISHERS, Netherlands. 2004, 81～88.

[16] Franklin M K, Reiter M K. Verifiable Signature Sharing. Advances in Cryptology—Eurocrypt'95, 1995, 50～63.

[17] Feng D G. Verifiable signature sharing for the DSA with heuristic secrecy, IEE Electronics letters, Vol. 32, No. 15, 1996, 1570～1571.

[18] Zheng Y. Digital Signcryption or How to Achieve Cost (Signature & Encryption) << Cost

(Signature) + Cost (Encryption). In: Burton S. Kaliski Jr. (Ed.). Advanced in Cryptology-CRYPTO'97, Annual International Cryptology Conferences, Proceedings, LNCS1294, Berlin: Springer, 1997, 165～179.
[19] 陈伟东,冯登国. 签密方案在分布式协议中的应用. 计算机学报. 2005. 28(9): 1421～1430.
[20] Gennaro R, Jarecki S, Krawczyk H. The security of distributed key generation in log-based Cryptosystems. In: Jacques (Ed.). Advances in Cryptology-Eurocrypt'99, LNCS 1592, Berlin: Springer-Verlag, 1999, 295～310.
[21] Laih C S, Yen S M. Multisignature for specifical group of verifier. Journal of Information Science and Engineering, 1996. 12(1): 143～152.
[22] He W H. Weakness in some multisignature schemes for specified group of verifiers. Information Procesing Letters, 2002. Vol. 83: 95～99.
[23] Eu Jin Goh, Stanislaw Jarecki. A Signature Scheme as Secure as the Diffie-Hellman Problem. In: Biham E, ed. Advances in Cryptology-EUROCRYPT'03, LNCS2656. Berlin: Springer-Verlag Press, 2003, 401～415.
[24] 陈伟东,冯登国. 指定验证方的门限验证签名方案及安全性证明. 软件学报. 2005. 16(11): 1967～1974.
[25] Rivest A. Shamir, Tauman Y. How to leak a secret. Advances in Cryptology-proc. Of ASIACRYPT'01, LNCS 2248, Springer-Verlag, 2001, 552～565.
[26] Kudla J. Special signature scheme and key agreement protocols. Thesis submitted to the university of London for the degree of doctor of philosophy. Information security group department of mathematics Royal Hollway, university of London, 2006.
[27] Abe M, Ohkub M, Suzuki K. 1-out-of-n signatures from a variety of keys. In: Y. Zheng, editor, Advances in Cryptology-proc. Of ASIACRYPT'02, LNCS 2501, Springer-Verlag, 2002, 415～432.
[28] Schnorr C. Efficient signature generation by smart cards. J. Cryptology, 1991. 4(3): 161～174.
[29] Pointcheval D, Stern J. Security arguments for digital signatures and blind signatures. Journal of Cryptology, vol. 13, 361～396, 2000.
[30] Jakobsson M, Sako K, Impagliazzo R. Designated verifiers proofs and their applications. Advances in Cryptology(Proccedings of Eurocrypt'96), LNCS 1070, Springer-Verlag. 1996, 143～154.
[31] Saeednia S, Kremer S, Oliver Markowitch. An efficient strong designated verifier signature scheme. In: Lim J and Lee D, Editors, Information Security and Cryptology-ICISC 2003, volume 2971 of Lecture Notes in Computer Science, 40～54. Springer-Verlag, 2003.
[32] Susilo W, Zhang F, Mu Y. Identity-based strong designated verifier signature scheme. In: H. Wang et al., Editors, ACISP 2004, volume 3108 of Lecture Notes in Computer Science, 313～324. Springer-Verlag, 2004.
[33] Seon J, Chang J H. Strong designated verifier proof signature without hash function and the same scheme for an Ad-hoc group ring. International Journal of Computer Science and Network Security, Vol. 6, No. 12, 2006, 205～210.
[34] Bellare M. Phillip Rogaway. Random Oracles are practical: A Paradigm for Designing Efficient Protocols. In: the Proceedings of the First ACM Conference on Computer and Communicatuions Security. New York: ACM Press, 1993. 62～73.
[35] Okamoto T, Pointcheval D. The gap-problems: a new class of problems for the security of cryptographic schemes. In K. Kim, editor, Public Key Cryptography-PKC 2001, volume 1992 of Lecture Notes in Computer Science, pages 104～118. Springer-Verlag, 2001.

[36] Nyberg K, Rueppel R A. A new signature scheme based on the DSA giving message recovery. 1st. ACM Conferences on Computer and Communications Security, Nov 3～5,1993, Fairfax,Virginia.

[37] Nyberg K, Rueppel RA. Message recovery for signature schemes based on the discrete logarithm problem. In: De Santis A, ed. Advances in Cryptology-EUROCRYPT'94. Lecture Notes in Computer Science 950, Berlin: Springer-Verlag, 1995. 182～193.

[38] Mario. Deniable authentication and key exchange. 2006.

[39] Bellare M. Provably Secure Session Key distribution—The Three Party case. In Proceedings of the ACM Symposium on the Theory of Computing, May,1995.

[40] Krawczyk H. HMQV: a high-performance secure Diffie-Hellman protocol.

[41] 冯登国,陈伟东. Generic Construction Methods of Strong Designated Verifier Signature Schemes,信息安全国家重点实验室第四届安全协议研讨会论文集,2009,27～45.

[42] Mitchell C J, Piper F, Wild P. Digital signatures. In Contemporary Cryptology, The Science of Information Integrity,325～378. IEEE Press, 1992.

[43] Pedersen T P. Signing contracts and paying electronically. Lecture Notes in Computer Science, 1561 (1999), 134～157. (Lectures on Data Security.)

[44] 冯登国. 数字签名技术概述,通信保密,No. 3, 1996,15～22.

[45] Zhang Z F, Feng D G. Cryptanalysis of some signature schemes with message recovery. Applied Mathematics and Computation 170(1): 103～114 (2005).

[46] Xu J, Zhang Z F, Feng D G. A Ring Signature Scheme Using Bilinear Pairings. WISA 2004: 160～169.

[47] Xu J, Zhang Z F, Feng D G. ID-Based Aggregate Signatures from Bilinear Pairings. CANS 2005: 110～119.

[48] Wang H, Qiu G, Feng D G. Cryptanalysis of Tzeng-Tzeng Forward-Secure Signature Schemes. IEICE Transactions 89-A(3): 822～825 (2006).

[49] Wang H, Zhang Y Q, Feng D G. Short Threshold Signature Schemes Without Random Oracles. INDOCRYPT 2005: 297～310.

[50] Lü X,Feng D G. An Arbitrated Quantum Message Signature Scheme. CIS 2004: 1054～1060.

第 9 章　身份识别协议

简单地讲，身份识别协议的目标就是使某人的身份被确认。假定你想向其他人证明你的身份，可以通过证明“你是什么”、“你有什么”和“你知道什么”这 3 种方式中的一种或几种来完成。“你是什么”是指你的行为或物理属性；“你有什么”是指文件或信用证明；“你知道什么”是指口令或个人信息等。

一个安全的身份识别协议至少应满足以下两个条件。

(1) 证明者 A 能向验证者 B 证明他的确是 A。

(2) 在证明者 A 向验证者 B 证明他的身份之后，验证者 B 不能获得关于 A 的任何有用的信息使得他能模仿 A 向第三方证明他是 A。

这两个条件是说证明者 A 能向验证者 B 电子地证明他的身份，而又没有向 B 泄露他的识别信息。目前已设计出许多满足这两个条件的身份识别协议。

身份识别协议与数字签名算法有着密不可分的联系。一方面，可以基于数字签名算法或 MAC 算法来构建安全的身份识别协议；另一方面，也可以直接针对身份识别协议的安全要求设计身份识别协议，并可通过一定的方式转化为数字签名算法。本章将从这两个方面介绍一些比较流行的、有代表性的身份识别协议。

9.1　基于 MAC 算法的身份识别协议

9.1.1　挑战-响应协议

首先看一个很简单但不安全的身份识别协议，它可基于任何 MAC 算法(如1.4.2 小节讨论的 MAC 算法)构建，也被称为挑战-响应协议，详见协议 9.1。这里假定 A 向 B 来识别自己，他们的共同密钥为 K，用消息认证码 MAC_K 来计算认证标签。

协议 9.1　不安全的挑战-响应协议。

(1) B 随机选择挑战 r，并发送给 A。

(2) A 计算 $y=\mathrm{MAC}_K(r)$，并发送给 B。

(3) B 计算 $y'=\mathrm{MAC}_K(r)$，如果 $y'=y$，则 B“接受”；否则，B“拒绝”。

这里首先解释一下交互协议中经常用到的会话和流这两个术语。一般而言，一个交互协议包括彼此通信的两方或多方。每一方交替地发送和接收消息。每运行一次协议称为一个会话。在会话中的每一步称为流，一个流包括消息从一方传给另一方。协议 9.1 包括两个流，第一个消息流从 B 传给 A，第二个消息流从 A 传给 B。会话结束时，B(会话的发起者)“接受”或“拒绝”(这是 B 在会话结束时的内部状态)，对 A 来说可能并不知道 B 是接受还是拒绝。

即便所使用的 MAC 算法是安全的，协议 9.1 也不能抵抗一种典型的攻击——并行会话攻击，因此它是不安全的。在图 9.1 中描述了这种攻击，敌手 O 可以成功地冒充 A。

在第一个会话进行中(假定 O 正在向 B 冒充 A),O 发起第二个会话,他主动让 B 来识别自己。第二个会话在图 9.1 中方框里描述。在第二个会话中,O 把在第一个会话中从 B 传来的挑战发送给 B。一旦他收到 B 的响应,O 继续第一个会话,他把 B 的响应发送回 B。这样 O 就成功地完成了第一个会话。

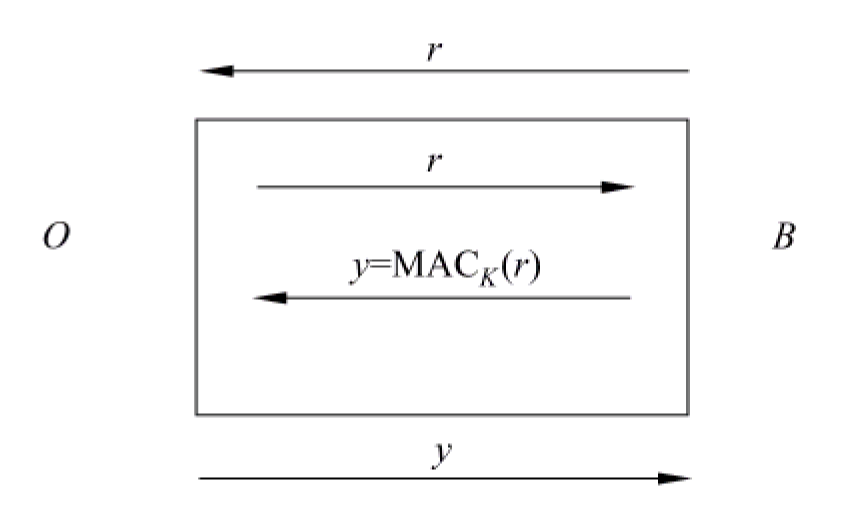


图 9.1　协议 9.1 的攻击过程示意图

并行会话攻击并不是在所有的应用场景中都构成真正的威胁,但设计身份识别协议时考虑其抵抗这种攻击是明智的。在下面的协议 9.2 中提出了一种简单的办法来改正这个缺陷。与协议 9.1 相比,仅有的改变是把身份标识符 ID 加入到 MAC 中来计算认证标签。

协议 9.2　安全的挑战-响应协议。

(1) B 随机选择挑战 r,并发送给 A。

(2) A 计算 $y=\mathrm{MAC}_K(\mathrm{ID}(A)\parallel r)$,并发送给 B。

(3) B 计算 $y'=\mathrm{MAC}_K(\mathrm{ID}(A)\parallel r)$,如果 $y'=y$,B“接受”;否则,B“拒绝”。

在协议 9.2 中,假定随机挑战是一指定长度的比特串,即 kb(如 $k=100$ 就是一种合适的选择)。也假定身份标识(ID(A)或 ID(B))是一特定长度的比特串,以某种标准固定的方式格式化。还假定身份标识串中包含了能区分网络中唯一实体的特征信息(因此 B 不必担心哪一个“A”正在与其谈话)。

协议 9.2 可以抵抗并行会话攻击。这是因为,如果 O 同样发起上述攻击,他会收到第二个会话中从 B 传来的 $\mathrm{MAC}_K(\mathrm{ID}(B)\parallel r)$。这对于 O 来说是没有帮助的,由于第一个会话中应该是值 $\mathrm{MAC}_K(\mathrm{ID}(A)\parallel r)$来响应 B 的挑战。

虽然协议 9.2 能够抵抗并行会话攻击,但并没给出能抵抗各种可能攻击的安全性证明。下面给出一个简短的安全性证明。

首先给出关于协议中的 3 个前提假设。

(1) 密钥假设。假定只有 A 和 B 知道密钥 K。

(2) 随机挑战消息假设。假定 A 和 B 都有理想的随机数生成器来产生他们的挑战消息。因此,在两个不同会话中产生相同挑战消息的概率是相当小的。

(3) MAC 安全假设。假定 MAC 算法是安全的。更精确地说,对于相应的 ε 和 Q,MAC 不存在(ε,Q)伪造。即当 O 最多知道 Q 个其他 MAC 值时,即 $\mathrm{MAC}_K(x_i)$,$i=1,2,\cdots,Q$,对于任意 i,$x\neq x_i$,O 能正确计算 $\mathrm{MAC}_K(x)$的概率最多为 ε。通常,Q 是指定的安全参数($Q=10000$ 或 100000 也许是合理的选择,具体由实际应用来决定)。

O 能够观察到 A 和 B 之间的多次会话。O 的目标是欺骗 A 或 B,即让 B“接受”实际没有 A 参与的会话,或让 A“接受”实际没有 B 参与的会话。要证明当上述 3 个假设都成立时,O 欺骗 A 或 B 的概率是相当小的。通过分析协议的结构,很容易完成证明。

假定 B“接受”,则 $y=\mathrm{MAC}_K(\mathrm{ID}(A)\parallel r)$,其中 y 是他在第二个流中收到的消息,r 是 B 在第一个流中发送的挑战。我们认为值 y 是由 A 根据挑战 r 产生的响应,这种情况发生的概率相当大。为了证明这种说法,让我们考虑响应不是直接来自 A 的可能情形。首先,由于假定密钥 K 只有 A 和 B 知道,不必考虑知道密钥 K 的其他人,计算 $y=\mathrm{MAC}_K(\mathrm{ID}(A)\parallel$

r)的概率。因此,或者 O 不知道密钥 K 而计算出了 y,或者值 y 由 A 或 B 在以前的会话中产生,而被 O 重用到本次会话中。

现在依次讨论这些可能的情形。

(1) 假定 $y=\text{MAC}_K(\text{ID}(A)\parallel r)$ 由 B 在以前的会话中产生。然而,B 只能计算 $\text{MAC}_K(\text{ID}(B)\parallel r)$,因此他不可能产生 y,这种情形不成立。

(2) 假定值 y 由 A 在以前的会话中产生。这种情形只有在挑战消息 r 被重用时才成立。然而,假定挑战消息 r 是由理想的随机数生成器产生,因此在不同会话中产生同样挑战消息的概率是相当小的。

(3) 假定值 y 是由 O 构造的新 MAC 值。由于 MAC 算法是安全的,O 不知道密钥 K,因此他产生 y 的概率是相当小的。

为了把上述非形式化的证明做一更精确的描述,先给出 MAC 算法安全的定义和身份识别协议安全的定义。

如果敌手最多查询了 Q 个消息的 MAC 值后,构造一个新消息的 MAC 值的概率不大于 ε(即不存在(ε,Q)伪造),就称 MAC 算法是无条件(ε,Q)安全的。通常,假定用固定密钥 K(敌手不知道 K)来构造 Q 个 MAC 值。如果敌手最多得到 A 和 B 之间的 Q 个会话消息后,成功欺骗 A 或 B 的概率不大于 ε,称身份识别协议是无条件(ε,Q)安全的。

对任何理想的 Q 和 ε,无条件(ε,Q)安全的 MAC 算法是确实存在的(如用强通用 Hash 族构造)。然而,无条件安全的 MAC 算法一般需要很长的密钥(尤其当 Q 很大时)。因此,像 CBC-MAC 算法这样的计算安全的 MAC 算法在实际中更常用一些。在这种情形下,MAC 算法的安全性假设是必要的。这种假设加入时间作为参数,其他模式与无条件安全假设类似。如果给定敌手的计算时间最多是 T,敌手最多查询了 Q 个消息的 MAC 值后,构造一个新消息的 MAC 值的概率不大于 ε,就称 MAC 算法是(ε,Q,T)安全的。如果给定敌手的计算时间最多是 T,敌手最多得到 A 和 B 之间的 Q 个会话信息后,成功欺骗 A 或 B 的概率不大于 ε,则称身份识别协议是(ε,Q,T)安全的。

为简化定义,通常省略时间参数。这样计算安全和无条件安全的定义就是类似的。至于使用的是计算安全还是无条件安全,通过上下文的内容就可以分辨清楚了。

现在来证明下面的定理。

定理 9.1 假定 MAC 算法是(ε,Q)安全的,假定随机挑战的长度是 kb。那么身份识别协议 9.2 是($Q/2^k+\varepsilon,Q$)安全的。

证明 首先假定基于无条件(ε,Q)安全的 MAC 算法构建身份识别协议,那么只要敌手在会话消息收集期间最多得到 Q 个有效的 MAC 值(使用相同的 MAC 密钥),最终的身份识别协议也是无条件安全的。还要考虑另外一个参数 k,也就是协议中随机挑战的比特长度。在这些条件下,能容易地给出敌手欺骗 B 的概率的上界。同样考虑以下 3 种情形。

(1) 正如上面所讨论,$y=\text{MAC}_K(\text{ID}(A)\parallel r)$不可能由 B 在以前的其他会话中产生。因此这种情形不可能发生。

(2) 假定值 y 由 A 在以前的会话中产生,挑战消息 r 是 B 新产生的,那么 B 在以前的会话中使用相同挑战 r 的概率是 $1/2^k$。由于最多可以考虑 Q 个以前的会话,因此 r 被重用的概率是 $Q/2^k$。如果 r 被重用的这种情形发生,敌手就可以重用以前会话中的 MAC 值。挑战 r 被重用的精确概率是 $1-(1-2^{-k})^Q$,小于 $Q/2^k$。

(3) 假定值 y 是由 O 构造的新 MAC 值。由于 MAC 算法是安全的，O 成功欺骗的概率最多是 ε。

综上所述，O 欺骗 B 的概率最多为 $Q/2^k+\varepsilon$。因此，建立了身份识别协议的安全性。

如果 MAC 算法是计算安全的，分析过程也基本上是一致的。

9.1.2 攻击模型和敌手目标

为了表述清楚，在图 9.2 中描述了中间入侵攻击的情形。

初看起来，这似乎是一个并行会话攻击。可以认为 O 在一个会话中向 B 冒充 A，在并行会话中向 A 冒充 B。当 O 收到 B 的挑战 r 后，他把 r 发送给 A。然后 O 把 A 的响应 y 发送给 B，B 将“接受”。然而，我们认为这不是一个真实的攻击，因为两个“会话”的“合成”是一个简单的会话，A 成功地向 B 识别自己的身份。全部的结果是 B 发起挑战 r，A 计算正确的响应 y。O 只是简单地转发消息，而没有修改消息，因此，O 在协议中不是一个主动的参与者。会话的实施过程就像 O 完全没有出现一样。

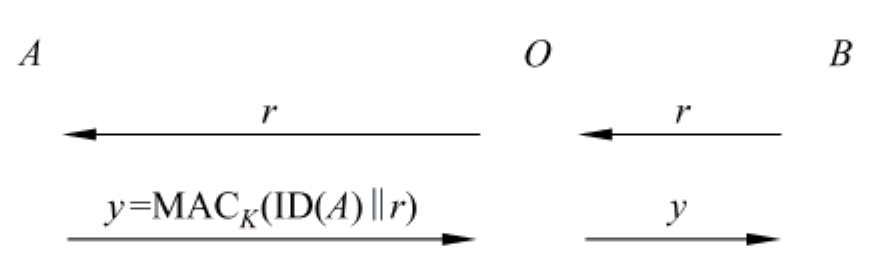


图 9.2 中间入侵攻击示意图

定义敌手 O 是主动的，如果以下条件之一成立：

(1) O 产生了一个新消息，并放入信道中。

(2) O 改变信道中的消息。

(3) O 转移信道中的消息，发送给其他人，而不是指定的接收者。

敌手的目标是主动的敌手能够让协议中的发起者(如 B，假定他是诚实的)在会话中“接受”。根据这个定义，O 在前面考虑的中间入侵攻击情形中，不是主动的敌手，因此敌手的目标是不能实现的。

判定敌手是否为主动敌手的另外一种等价方式是考虑协议中 A 和 B 的视图。A 和 B 都在与指定的实体通信：A 的指定实体是 B，反之亦然。进一步，如果没有主动敌手，那么 A 和 B 会有匹配的会话视图：A 发送的每个消息都是 B 所接收的；反之亦然。而且，没有乱序的消息被收到。这些会话特征被称为匹配会话(Matching Conversations)。

上述模型的讨论假定会话中的合法参与方都是诚实的。准确地说，如果会话中的参与方(如 A 或 B)严格按照协议流程执行，进行正确的计算，不向敌手(O)泄露任何信息，则被称为是诚实的参与方。如果参与方是不诚实的，那么协议就完全被攻破了。因此，安全假定通常要求参与方是诚实的。

现在再来考虑攻击模型。在 O 实际欺骗 B 之前，需要进行信息收集。如果他只是观察 A 和 B 之间的会话，O 在此过程中是一个被动敌手。也可以考虑 O 在信息收集阶段是主动敌手的攻击模型。例如，O 可以临时地访问预言器(Oracle)，来计算认证标签 $MAC_K(\cdot)$，密钥 K 为 A 和 B 共享(O 不知道 K)。在这一阶段，O 肯定能成功地欺骗 A 和 B，因为他可以利用预言器来响应挑战。然而，当信息收集阶段完成后，MAC 预言器就不能再使用，那么 O 在新的会话中实施他的攻击让 A 或 B“接受”，但不能访问 MAC 预言器。

9.1.1 小节中的安全性分析都可以应用到这些攻击模型中。如果 MAC 算法在已知消息攻击下是(ε,Q)安全的，那么身份识别协议在被动敌手信息收集模型下是可证明安全的(确切地说，敌手的成功概率最多为 $Q/2^k+\varepsilon$)。进一步，如果 MAC 算法在选择消息攻击下

是(ε,Q)安全的，那么身份识别协议在主动敌手信息收集模型下是安全的。

9.1.3 交互认证协议

A 和 B 都向对方证实各自的身份，这种协议称为交互认证协议或交互身份识别协议。会话成功完成时，参与双方都为“接受”状态。敌手试图欺骗 A 或 B 或双方，使其接受。敌手的目标是进行主动攻击后，使得诚实的参与方“接受”。

一个安全的交互认证协议应具备以下两个条件。

(1) 假定 A 和 B 是会话中的两个参与方，他们都是诚实的。也假定敌手是被动的，那么 A 和 B 都将“接受”。

(2) 如果敌手是主动的，则会话完成后，诚实的参与方都不会“接受”。

值得一提的是，在一个特定的会话中，敌手也许开始是被动的，当一方接受后，就变成主动的。因此一个诚实的参与方“接受”，而另一个诚实的参与方“拒绝”，这种情形是可能发生的。在这种情形下，虽然会话没有成功地完成，敌手也没有达到他的目标，因为敌手在第一个参与方接受之前是被动的。会话的结果是 A 成功地向 B 证实了自己的身份，而 B 没有成功地向 A 证实自己的身份。这可以看做是协议的中断，但却不是一个成功的攻击。

在交互认证协议的会话中，主动敌手有以下几种表现方式。

(1) 敌手伪造 A，希望 B 接受。

(2) 敌手伪造 B，希望 A 接受。

(3) 敌手在 A 和 B 参与的会话中是主动的，希望 A 和 B 都接受。

可以通过运行两次协议 9.2 来达到交互认证(通过两个独立的会话，让 A 验证 B 的身份，B 验证 A 的身份)。然而，设计一个简单的协议一次完成交互认证，应该是更高效的。

那么如何以简单的方式，把单向身份识别的两个会话合并为一个协议呢？协议 9.3 给出了一个解决方案，它把消息流从 4 个减为 3 个(与运行单向身份识别协议两次相比)。然而，最终的交互身份识别协议是存在缺陷的，并且易受攻击。

协议 9.3 不安全的交互认证协议。

(1) B 随机选择挑战 r_1，发送给 A。

(2) A 随机选择挑战 r_2，计算 $y_1=\mathrm{MAC}_K(\mathrm{ID}(A)\parallel r_1)$，发送 r_2 和 y_1 给 B。

(3) B 计算 $y_1'=\mathrm{MAC}_K(\mathrm{ID}(A)\parallel r_1)$，如果 $y_1'=y_1$，B“接受”；否则，B“拒绝”。B 也计算 $y_2=\mathrm{MAC}_K(\mathrm{ID}(B)\parallel r_2)$，发送 y_2 给 A。

(4) A 计算 $y_2'=\mathrm{MAC}_K(\mathrm{ID}(B)\parallel r_2)$，如果 $y_2'=y_2$，A“接受”；否则，A“拒绝”。

由于 O 在并行会话攻击中能够欺骗 A，协议 9.3 是不安全的。O 伪装成 B，发起一次与 A 的会话。当 O 在第二个消息流中收到 A 的挑战 r_2 时，他接受，然后发起与 B 的第二次会话(伪装成 A)。在第二次会话中，O 在第一个消息流中发送挑战 r_2 给 B。当 O 收到 B 的响应(第二次会话的第二个消息流)，他转发给 A 作为第一次会话的第三个消息流。A 会“接受”，因此 O 在这次会话中成功地冒充了 B。这构成了一个有效的攻击，因为第一次会话中诚实参与方(也就是 A)最终接受了主动敌手 O(O 在此次会话中发送了第一个挑战)。图 9.3 描述了详细的攻击过程。

显然，这种攻击是把一个会话中的消息重用到另一个会话的不同消息流中。弥补这个缺陷并不困难，实际上可有几种方式来改进此协议使其安全。最基本的设计思想是要保证

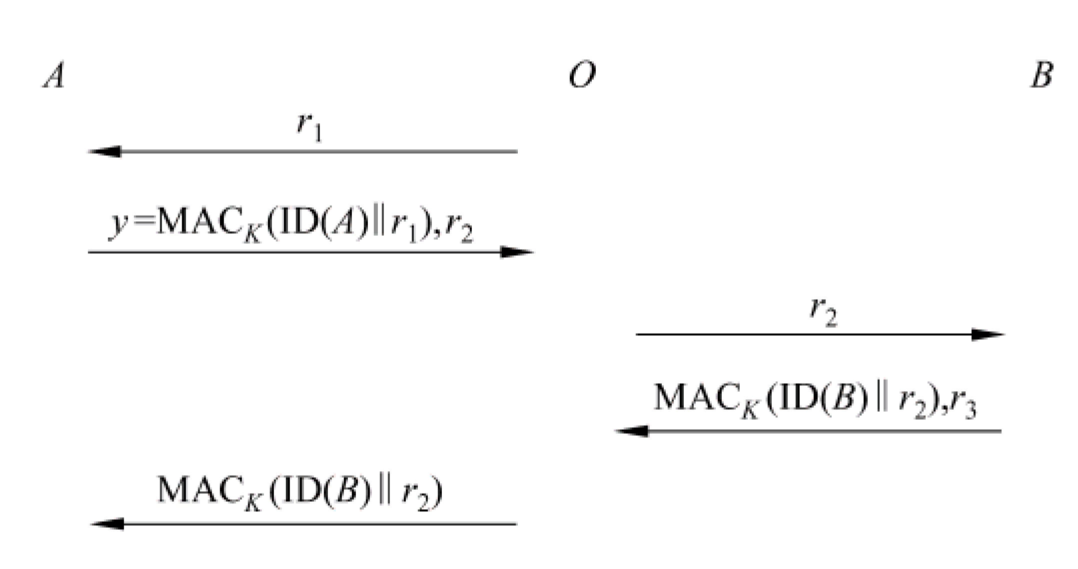


图 9.3　协议 9.3 的攻击过程示意图

每个消息流中的消息以不同的方式来计算。协议 9.4 描述了一种解决方案。

协议 9.4　安全的交互认证协议。

(1) B 随机选择挑战 r_1，并发送给 A。

(2) A 随机选择挑战 r_2，计算 $y_1=\text{MAC}_K(\text{ID}(A)\parallel r_1\parallel r_2)$，并发送 r_2 和 y_1 给 B。

(3) B 计算 $y_1'=\text{MAC}_K(\text{ID}(A)\parallel r_1\parallel r_2)$，如果 $y_1'=y_1$，B"接受"；否则，B"拒绝"。B 也计算 $y_2=\text{MAC}_K(\text{ID}(B)\parallel r_2)$，发送 y_2 给 A。

(4) A 计算 $y_2'=\text{MAC}_K(\text{ID}(B)\parallel r_2)$，如果 $y_2'=y_2$，A"接受"；否则，A"拒绝"。

在协议 9.4 中，只有第(2)步 y_1 的定义做了改变。MAC 的参数包含 r_1 和 r_2，这样第二个消息流和第三个消息流(MAC 的参数仅包含 r_2)中的消息有不同的计算方式。

协议 9.4 的安全性分析与协议 9.2 类似，但稍复杂一些，因为敌手或伪装成 B(欺骗 A)或伪装成 A(欺骗 B)。可以计算 y_1 或 y_2 是以前会话"重用"的概率，也可以计算敌手重新产生新 MAC 的概率。可证明以下的定理。

定理 9.2　假定 MAC 算法是(ε,Q)安全的，假定随机挑战的长度是 kb。那么交互身份识别协议 9.4 是$(Q/2^k+2\varepsilon,Q/2)$安全的。

证明　由于 y_1 和 y_2 的计算方式不同，一次会话中的 y_1 重用到另一次会话中的 y_2 是不可能的(反之亦然)。O 可通过得到 y_2 来伪装 B 欺骗 A，或得到 y_1 来伪装 A 欺骗 B。假设 O 最多可以从以前的会话中得到 Q 个 MAC 值(由于每次会话有 2 个 MAC 值，因此要限制 O 仅能查询 $Q/2$ 次以前的会话)，则 y_1 或 y_2 是以前会话重用的概率最多为 $Q/2^k$。O 能产生新的 y_1 的概率是 ε，能产生新的 y_2 的概率也是 ε。因此，O 欺骗 A 或 B 的概率最多为 $Q/2^k+2\varepsilon$。

9.2　基于数字签名算法的身份识别协议

假定 A 和 B 都是网络中的成员，针对特定的数字签名算法，每个参与方都有相应的公钥和私钥。在这种环境中，总是需要提供一种装置来证实网络中其他用户的公钥，这就需要某种公钥基础设施(PKI)。总之，假定有一个可信的授权机构(记为 TA)，由它来签署网络中所有用户的公钥(在 PKI 中，可信的授权机构常常称为证书认证中心，记为 CA，然而这里用 TA 来表示)，所有用户都知道 TA 的公开验证密钥 Ver_{TA}。

9.2.1　证书发放协议

网络用户的证书包含用户的身份信息(如他的名字、E-mail 地址等)、公钥以及 TA 对这

些信息的签名。证书允许网络用户验证彼此公钥的真实性。例如，假定 A 想从 TA 处获得证书，同时包含 A 的公钥的副本，将执行以下给 A 发放证书的协议。

(1) TA 通过出生证或护照等身份证明来确定 A 的身份 ID(A)，ID(A)中包含 A 的身份信息。

(2) A 的私有签名算法 Sig_A 和相应的验证算法 Ver_A 被确定。

(3) TA 产生对 A 的身份标识和验证算法的签名 $s=\mathrm{Sig}_{TA}(\mathrm{ID}(A)\parallel \mathrm{Ver}_A)$，把证书 $\mathrm{Cert}(A)=(\mathrm{ID}(A)\parallel \mathrm{Ver}_A\parallel s)$连同 A 的私有签名算法 Sig_A 一起发送给 A。

这里并不去详细地说明 A 怎样向 TA 识别自己、ID(A)的格式以及 A 的公钥和私钥的选取。一般来说，这些具体的应用细节由 PKI 来决定。

任何知道 TA 的验证算法 Ver_{TA}的用户都可以验证其他用户的证书。假定 B 想要确认 A 的公钥的真实性。A 把证书给 B，B 就可通过下面的等式来验证 TA 的签名：

$$\mathrm{Ver}_{TA}(\mathrm{ID}(A)\parallel \mathrm{Ver}_A, s) = \mathrm{true}$$

证书的安全性直接由 TA 所使用的签名算法的安全性得到保障。

验证证书的目的是认证用户公钥的真实性。证书本身并不提供任何的身份证明，因为证书仅仅包含一些公共信息。证书能发布或重新发布给任何用户，证书的拥有并不意味着独有。

9.2.2　基于数字签名算法的身份识别协议

我们的基本想法是用数字签名算法取代协议 9.4 中的 MAC 算法。另一个不同之处是，在协议 9.4 中，每个 MAC 的计算要包含用户(产生此 MAC 的用户)的名字(这是很重要的，因为密钥 K 是双方共享的，任何一方都有可能产生 MAC)。而在这里，仅仅一方，也就是拥有私钥的那一方，才能产生用此私钥作的签名。因此，就没有必要清晰地指出产生此签名的用户。

与协议 9.4 一样，在会话的开始，每个参与方认定一个与其通信的指定用户，然后利用指定用户的验证算法来验证会话中收到的签名信息。所有的签名信息中都要包含指定的通信用户(接收签名的用户)的名字。

下面介绍一个典型的基于数字签名算法的交互身份识别协议[1]。只要数字签名算法是安全的，挑战是随机产生的，就可以证明这个身份识别协议是安全的。具体识别过程如下。

(1) B 随机选择挑战 r_1，发送 Cert(B)和 r_1 给 A。

(2) A 随机选择挑战 r_2，计算 $y_1=\mathrm{Sig}_A(\mathrm{ID}(B)\parallel r_1\parallel r_2)$，发送 Cert($A$)，$r_2$ 和 y_1 给 B。

(3) B 利用证书 Cert(A)验证 A 的公钥 Ver_A。然后验证 $\mathrm{Ver}_A(\mathrm{ID}(B)\parallel r_1\parallel r_2, y_1)=\mathrm{true}$ 是否成立。如果成立，则 B“接受”；否则，B“拒绝”。B 也计算 $y_2=\mathrm{Sig}_B(\mathrm{ID}(A)\parallel r_2)$，发送 y_2 给 A。

(4) A 利用证书 Cert(B)验证 B 的公钥 Ver_B。然后验证 $\mathrm{Ver}_B(\mathrm{ID}(A)\parallel r_2, y_2)=\mathrm{true}$ 是否成立。如果成立，则 A“接受”；否则，A“拒绝”。

只要上述协议使用的签名算法是安全的(签名算法的安全性定义类似于 MAC 算法的安全性定义)，则下面的定理说明了该协议是安全的。

定理 9.3　假定签名算法 Sig 是(ε, Q)安全的，假定随机挑战的长度是 kb。那么上述交互身份识别协议是$(Q/2^{k-1}+2\varepsilon, Q)$安全的。

值得一提的是，在定理9.3中，可查询的以前会话的次数是Q，而定理9.2中却被限制为$Q/2$。这是因为该协议中A和B使用不同的密钥产生签名。敌手允许查询由A产生的Q个签名和B产生的Q个签名。与之相反，协议9.4中A和B使用相同的密钥产生MAC。由于对给定的密钥，限制敌手最多查询Q个由此密钥产生的MAC，因此这就迫使要求敌手仅能查询$Q/2$个以前的会话。

9.3 Feige-Fiat-Shamir 身份识别协议

前面两节介绍了利用密码学工具设计身份识别协议的方法，设计身份识别协议的另一种方法是“从零开始”构建，无须使用任何密码学工具。这种方法设计的协议与前面的协议相比，一个潜在优势是他们可能更高效、具有更低的通信复杂度。这种协议一般通过向他人证明自己知道某些秘密值（如私钥），而不泄露这个秘密值的方式来证实自己的身份。在剩下的章节中主要介绍通过这种方法设计的协议。

Fiat 和 Shamir 在1986年基于零知识证明思想提出了一种新型的身份识别协议[2]，后经 Feige、Fiat 和 Shamir 改进成为身份的零知识证明[3]。本节重点描述 Feige-Fiat-Shamir 身份识别协议，该协议的目的是能使证明者A向验证者B证明他的身份，而事后B又不能冒充A。这里假定存在一个可信中心，该中心的唯一目的是秘密地选取形式为$4r+3$的两个大素数（这种整数称为 Blum 整数）使得$n=pq$是计算上难分解的，然后公布n作为所有用户的模。在公开n后，中心可以被取消或关闭，因为它再没有其他作用。

A的秘密身份证$i(A)$的产生过程如下。

(1) 在Z_n中随机选择k个数$S_1,S_2,\cdots,S_k$。

(2) 随机地，独立地选择$I_j=1/S_j^2 \bmod n$或$-1/S_j^2 \bmod n(1\leqslant j\leqslant k)$。

(3) 公开$I_1,I_2,\cdots,I_k$，将$P_i(A)=(I_1,I_2,\cdots,I_k)$作为$A$的公开身份证；保密$S_1,S_2,\cdots,S_k$，将$i(A)=(S_1,S_2,\cdots,S_k)$作为$A$的秘密身份证。

验证者B知道公开的n和$P_i(A)$，证明者A想使B相信他知道$i(A)$，但又不想对B泄露任何信息。为了达到这一目的，A和B重复执行下列步骤t次（轮数t能降低A的欺骗概率）。

(1) A选择一个随机数R，计算$X=R^2 \bmod n$或$-R^2 \bmod n$，并将X发送给B。

(2) B随机选择一个向量$(E_1,\cdots,E_k)\in Z_2^k$，并发送给$A$。

(3) A计算$Y=R\prod\limits_{\substack{E_j=1\\1\leqslant j\leqslant k}} S_j \bmod n$，并将$Y$发送给$B$。

(4) B验证是否有$X=\pm Y^2\prod\limits_{\substack{E_j=1\\1\leqslant j\leqslant n}} I_j \bmod n$。

在上述身份识别过程中，A没有提交他的秘密身份证，也没有向B证明他的公开身份证的合法性，而是向B通过显示他拥有关于他的秘密身份证的知识来证明他的公开身份证的合法性。

身份识别与数字签名的目的不同，身份识别主要用于实体的身份认证，而数字签名主要用于消息的完整性认证或消息的源和目的认证。但每一个身份识别协议都可以派生出一个数字签名算法，其标准做法是用一个公开的Hash函数来代替身份识别协议中的验证者B。

在以下的章节中每介绍一个身份识别协议将相应地介绍一个数字签名算法。下面以 Feige-Fiat-Shamir 身份识别协议为例，来说明如何将一个身份识别协议转化为一个数字签名算法，将后者称为 Feige-Shamir 数字签名算法，又称 Feige-Shamir 启发式签名算法，后来人们也把这类签名称为知识签名[4]。

用户 A 随机选取一个模数 n，n 为两个大素数之积。实际应用中，n 至少为 512b 长，可能接近 1024b。用户 A 选取 k 个不同的数 $v_1,v_2,\cdots,v_k$，这里 v_i 为模 n 的平方剩余。计算 $s_i=1/v_i^{\frac{1}{2}} \bmod n$，$1\leqslant i\leqslant k$。签名者 A 的公钥为 $v_1,v_2,\cdots,v_k$，并将其在可信中心注册，私钥为 $s_1,s_2,\cdots,s_k$。设 A 使用的 Hash 函数为 $H(\cdot)$，$H(\cdot)$是公开的。

A 对消息 m 的签名过程如下。

(1) A 在$[0,n)$中随机选择 t 个整数 $r_i(1\leqslant i\leqslant t)$，并计算 $x_i=r_i^2 \bmod n$。

(2) A 计算 $H(m\parallel x_1\parallel x_2\parallel\cdots\parallel x_t)$，用 $e_{ij}(1\leqslant i\leqslant t,\ 1\leqslant j\leqslant k)$表示 $H(m\parallel x_1\parallel x_2\parallel\cdots\parallel x_t)$的前 kt 比特。

(3) A 计算 $y_i=r_i\prod\limits_{e_{ij}=1}s_j \bmod n\ (1\leqslant i\leqslant t)$，并将 $m,y_i(1\leqslant i\leqslant t)$和 $e_{ij}(1\leqslant i\leqslant t,\ 1\leqslant j\leqslant k)$发给接收者 B。

B 验证 A 对 m 的签名的过程如下。

(1) B 计算 $z_i=y_i^2\prod\limits_{e_{ij}=1}v_j \bmod n, 1\leqslant i\leqslant t$。

(2) B 验证 $H(m\parallel z_1\parallel z_2\parallel\cdots\parallel z_k)$的前 kt 个比特是否与 $e_{ij}(1\leqslant i\leqslant t,\ 1\leqslant j\leqslant k)$一致。

Feige-Shamir 数字签名算法与 RSA 数字签名算法相比，主要优点是运算速度快，其中模乘法次数仅为 RSA 中的 1%～4%。kt 的大小被建议取在 20～72 之间，如 $k=9$、$t=8$。

9.4　Schnorr 身份识别协议

Schnorr 身份识别协议[5]融合了几种身份识别协议的思想，主要有 ElGamal 签名算法、Fiat-Shamir 身份识别协议和 Chaum-Evertse-Van de Graff 交互式协议等，其安全性建立在计算离散对数问题的困难性之上。

Schnorr 身份识别协议需要一个可信中心，记为 TA。TA 选择下列参数。

(1) p 是一个大素数($p\geqslant 2^{1024}$)，在 Z_p^* 上计算离散对数是难处理的。

(2) q 是一个大素数($q\geqslant 2^{160}$)，并且 $q\mid(p-1)$。

(3) $\alpha\in Z_p^*$，阶为 q(如可取 $\alpha=g^{(p-1)/q}$，g 为 Z_p 的本原元)。

(4) 一个安全参数 t，$2^t<q$(对大多数应用来说，取 $t=40$ 将已提供足够的安全性，为了更高的安全性，Schnorr 建议使用 $t=72$)。

(5) TA 选择一个安全的签名方案，记签名算法为 $\mathrm{Sig}_{\mathrm{TA}}$，验证算法为 $\mathrm{Ver}_{\mathrm{TA}}$。

(6) 选定一个安全的 Hash 函数。像通常一样，所有的信息在签名之前先进行杂凑，为了便于阅读，在描述协议时将略去杂凑这一步。

参数 p、q、α、$\mathrm{Ver}_{\mathrm{TA}}$和 Hash 函数都是公开的。

TA 给每个用户颁布一个证书。当 A 想从 TA 那里获得证书时，A 和 TA 执行下列

协议。

(1) TA 给申请者 A 建立并颁布一个标识串 ID(A),ID(A)包含 A 的足够多的信息,如姓名、性别、生日、职业、电话号码、指纹信息、DNA 码等识别信息。

(2) A 秘密地选择一个随机指数 a,$0\leqslant a\leqslant q-1$,计算 $v=\alpha^{-a} \bmod p$ 并将 v 发送给 TA。

(3) TA 对(ID(A),v)签名,$s=\mathrm{Sig}_{TA}(\mathrm{ID}(A),v)$。TA 将证书 $C(A)=(\mathrm{ID}(A),v,s)$发送给 A。

证明者 A 向验证者 B 证明他的身份的协议,即 Schnorr 身份识别协议可描述如下。

(1) A 随机选择一个数 k,$0\leqslant k\leqslant q-1$,计算 $\gamma=\alpha^k \bmod p$。

(2) A 把他的证书 $C(A)=(\mathrm{ID}(A),v,s)$和 γ 发送给 B。

(3) B 通过检查 $\mathrm{Ver}_{TA}(\mathrm{ID}(A),v,s)$是否为真来验证 TA 的签名。

(4) B 随机选择一个数 r,$1\leqslant r\leqslant 2^t$,并将 r 发送给 A。

(5) A 计算 $y=(k+ar) \bmod q$,并将 y 发送给 B。

(6) B 通过检验 $\gamma=\alpha^y v^r \bmod p$ 是否成立来识别 A,只要等式成立,B 就承认 A 的身份。

下面首先对 Schnorr 身份识别协议作一些解释。第(1)步可进行预处理,即在 B 出现之前完成。设置安全参数 t 的目的是阻止冒充者 C 伪装成 A 猜测 B 的挑战 r。因为如果 t 不够大,C 有可能事先猜测到 r 的正确值,那么 C 在第(1)步任取 y,计算 $\gamma=\alpha^y v^r \bmod p$,当他收到 B 发送来的挑战 r 时,他将已选好的 y 提供给 B,那么 y 和 γ 必能通过第(6)步 B 的验证。将把 γ 发送给 B,如果 B 随机地猜测 r,那么 C 能猜中的概率是 2^{-t}。这样,对大部分应用来说,$t=40$ 将是一个合理的选择。

签名 s 用来证明 A 的证书的合法性。当 B 验证了 TA 对 A 的证书的签名,他自己就相信证书本身是真实的。A 秘密选择的值 a 功能上类似于个人识别号 PIN,它使 B 相信完成识别协议的人的确是 A。但它与 PIN 有着本质的差别:在识别协议中,a 的值一直没有被泄露。而是 A(更精确地说是,A 的智能卡)向 B 证明他知道 a 的值。这一证明过程在识别协议的第(5)步完成,它通过 A 用计算值 y 响应 B 颁布的挑战 r 来完成。

现在来看 Schnorr 协议的安全性。

首先,冒充者 C 通过伪造一个证书 $C'(A)=(\mathrm{ID}(A),v',s')$, $v\neq v'$,来模仿 A 是难以成功的,因为这里的 s' 必须是 TA 对(ID(A),v')的签名,才能通过协议第(3)步中 B 的验证。但只要 TA 的签名方案是安全的,C 就不能伪造 TA 的这个签名 s'。

其次,C 改用 A 的正确证书 $C(A)=(\mathrm{ID}(A),v,s)$(证书不保密,是公开的)来模仿 A 也是难以成功的。因为这时他必须猜出 A 的密钥 a,才能在第(5)步计算出 $y=(k+ar) \bmod q$ 响应 B 提出的挑战 r。但是求 a 涉及求离散对数问题,而已假定在 Z_p 上计算离散对数是不可行的。

尽管如此,到目前为止,仍然没有证明 Schnorr 协议是安全的。不过,它的一个修改,即下节将要介绍的 Okamoto 识别协议可证明是安全的。

Schnorr 识别协议从计算量和需要交换的信息量两方面来看都是很快的和有效的。它也极小化了由 A 所完成的计算量。这是考虑到在许多实际应用中,A 的计算将由一个低计算能力的 Smart 卡来完成,而 B 的计算将由一个具有较强计算能力的计算机来完成。

为了说明问题,假定 ID(A)和 v 的长度均为 512b。如果 TA 使用 DSA 签名,那么 s 的长度为 320b。此时,证书 $C(A)$的总长度为 1344b。

让我们考虑一下 A 的计算量：第(1)步需要完成一个模指数运算；第(5)步需要一个模加法运算和一个模乘法运算。只有模指数运算是复杂的，需要的时间较长，但这个可以离线预计算。由 A 完成的在线运算是适中的。

Schnorr 签名算法：

设 p 是一个大素数，在 Z_p 上的离散对数问题是难处理的，q 也是一个大素数且 $q\mid(p-1)$。设 $\alpha\in Z_p^*$ 是一个阶为 q 的元素，H 是一个 Hash 函数。$P=Z_p^*$，$A=Z_p^*\times Z_q$，$K=\{(p,q,\alpha,a,v):v=\alpha^{-a} \bmod p\}$。值 p、q、α、v 和 H 是公开的，a 是保密的。对 $K=(p,q,\alpha,a,v)$ 和一个(秘密的)随机数 $k\in Z_q^*$，定义 $\mathrm{Sig}_K(x,k)=(\gamma,y)$，其中 $\gamma=\alpha^k \bmod p$，$y=(k+aH(x,\gamma)) \bmod q$。对 $x,\gamma\in Z_p^*$ 和 $y\in Z_q$，定义 $\mathrm{Ver}_K(x,\gamma,y)=$真$\Leftrightarrow\gamma=\alpha^y v^{H(x,\gamma)} \bmod p$。

9.5 Okamoto 身份识别协议

Okamoto 身份识别协议[6]是 Schnorr 身份识别协议的一种改进。在假定 Z_p 上计算一个特定的离散对数是难处理的情况下，可证明这种改进的安全性。但仅就速度和有效性来讲，Schnorr 协议比 Okamoto 协议更实用。

为了建立这一协议，TA 像在 Schnorr 协议中一样，选择两个大素数 p 和 q。TA 也在 Z_p 中选择两个阶为 q 的元素 α_1 和 α_2。对系统的所有参加者包括 A，TA 保密 $c=\log_{\alpha_1}\alpha_2$。假定任何人(即使 A 和 C 合伙)计算值 c 是不可行的。像在 Schnorr 识别协议中一样，TA 选择一个签名算法和一个 Hash 函数。

TA 向 A 颁布证书的协议如下。

(1) TA 给申请者 A 建立并颁布一个标识串 ID(A)。

(2) A 秘密地选择两个随机指数 a_1 和 a_2，$0\leqslant a_1,a_2\leqslant q-1$，计算 $v=\alpha_1^{-a_1}\alpha_2^{-a_2} \bmod p$ 并将 v 发送给 TA。

(3) TA 对(ID(A)，v)签名，$s=\mathrm{Sig}_{\mathrm{TA}}(\mathrm{ID}(A),v)$。TA 将证书 $C(A)=(\mathrm{ID}(A),v,s)$发送给 A。

Okamoto 身份识别协议如下。

(1) A 随机选择两个数 k_1 和 k_2，$0\leqslant k_1,k_2\leqslant q-1$，并计算 $\gamma=\alpha_1^{k_1}\alpha_2^{k_2} \bmod p$。

(2) A 将他的证书 $C(A)=(\mathrm{ID}(A),v,s)$和 γ 发送给 B。

(3) B 通过检验 $\mathrm{Ver}_{\mathrm{TA}}(\mathrm{ID}(A),v,s)$是否为真来验证 TA 的签名。

(4) B 随机选择一个数 r，$1\leqslant r\leqslant 2^t$，并将 r 发送给 A。

(5) A 计算 $y_1=(k_1+a_1r) \bmod q$ 和 $y_2=(k_2+a_2r) \bmod q$，并将 y_1、y_2 发送给 B。

(6) B 通过检验 $\gamma=\alpha_1^{y_1}\alpha_2^{y_2}v^r \bmod p$ 来识别 A。

Okamoto 协议和 Schnorr 协议的主要差别在于：在假定计算离散对数 $\log_{\alpha_1}\alpha_2$ 是不可行的情况下，能够证明 Okamoto 协议是安全的。

Okamoto 协议的完备性(也就是，B 接受 A 的身份证明)的证明是明显的。

安全性证明是很微妙的。总体思路是：假定 A 通过多项式次数执行协议向 O 识别自己的身份：假定(希望得到矛盾结果)O 能够获取关于 A 的秘密指数 a_1 和 a_2 的一些信息。如果此条件成立，可以证明 A 与 O 联合能在多项式时间内(以相当大的概率)计算出离散对

数 c。这就与上面提到的假设相矛盾，从而证明了 O 不能通过参与协议而获取关于 A 指数的任何信息。

证明过程的第一步类似于 Schnorr 身份识别协议的证明。假定 O 知道 γ，同时能计算两个不同挑战 r 和 s 的有效响应。也就是，假定 O 能计算 y_1、y_2、z_1、z_2、r 和 s，满足 $r \neq s$，而且

$$\gamma \equiv \alpha_1^{y_1} \alpha_2^{y_2} v^r \equiv \alpha_1^{z_1} \alpha_2^{z_2} v^s \pmod p$$

O 能计算

$$b_1 = (y_1 - z_1)(r - s)^{-1} \bmod q \quad 和 \quad b_2 = (y_2 - z_2)(r - s)^{-1} \bmod q$$

满足等式：$v \equiv \alpha_1^{-b_1} \alpha_2^{-b_2} \pmod p$。

现在证明 A 与 O 合谋能够计算 c（以相当大的概率）。假定以上所述 O 能够确定 b_1 和 b_2 满足

$$v \equiv \alpha_1^{-b_1} \alpha_2^{-b_2} \pmod p$$

那么假定 A 向 O 泄露秘密值 a_1 和 a_2。当然

$$v \equiv \alpha_1^{-a_1} \alpha_2^{-a_2} \pmod p$$

因此

$$\alpha_1^{a_1 - b_1} \equiv \alpha_2^{b_2 - a_2} \pmod p$$

假定 $(a_1, a_2) \neq (b_1, b_2)$，那么 $(b_2 - a_2)^{-1} \bmod q$ 存在，而且多项式时间内能计算离散对数

$$c = \log_{\alpha_1} \alpha_2 = (a_1 - b_1)(b_2 - a_2)^{-1} \bmod q$$

下面考虑 $(a_1, a_2) = (b_1, b_2)$ 的可能性。如果这种情形发生，c 就不能按上述公式被计算出来。然而，认为 $(a_1, a_2) = (b_1, b_2)$ 成立的概率只有 $\frac{1}{q}$，因此几乎能保证 A 与 O 合谋计算出 c。

定义

$$\mathcal{A} = \{(\alpha_1', \alpha_2') \in \mathbb{Z}_q \times \mathbb{Z}_q : \alpha_1^{-\alpha_1'} \alpha_2^{-\alpha_2'} \equiv \alpha_1^{-a_1} \alpha_2^{-a_2} \pmod p\}$$

也就是，$\mathcal{A}$ 包含了所有可能是 A 的秘密指数的有序对。注意到 $\mathcal{A}$ 可以表示为

$$\mathcal{A} = \{(a_1 - c\theta, a_2 + \theta) : \theta \in \mathbb{Z}_q\}$$

这里 $c = \log_{\alpha_1} \alpha_2$。因此 $\mathcal{A}$ 包含了 q 个有序对。

O 计算的有序对 (b_1, b_2) 一定在集合 $\mathcal{A}$ 中。认为有序对 (b_1, b_2) 中的值与 A 秘密指数 (a_1, a_2) 的值是独立的。由于 (a_1, a_2) 由 A 随机选择，因此 $(a_1, a_2) = (b_1, b_2)$ 成立的概率是 $\frac{1}{q}$。

需要解释一下 (b_1, b_2)"独立"于 (a_1, a_2) 的含义。A 的秘密指数 (a_1, a_2) 是集合 $\mathcal{A}$ 中 q 个可能的有序对之一，究竟哪一个才是"正确的"有序对，A 没有泄露任何信息。不严格地说，O 知道 A 的秘密指数对来自于 $\mathcal{A}$，但没有办法分辨是哪一个。

现在来看看身份识别协议执行过程中交换的信息。基本上，每次协议执行时，A 选择 γ；O 选择 r；然后 A 计算 y_1 和 y_2 满足

$$\gamma \equiv \alpha_1^{y_1} \alpha_2^{y_2} v^r \pmod p$$

A 具体计算

$$y_1 = (k_1 + a_1 r) \bmod q \quad 和 \quad y_2 = (k_2 + a_2 r) \bmod q$$

这里 $\gamma=\alpha_1^{k_1}\alpha_2^{k_2} \bmod p$。但要注意 k_1 和 k_2 没有公开(a_1 和 a_2 也没有公开)。

由于 y_1 和 y_2 的计算依赖于 a_1 和 a_2,协议执行过程中产生的四元组(γ,r,y_1,y_2)似乎依赖于 A 的有序对(a_1,a_2)。然而可以证明每个这样的四元组都能由任何其他的有序对(a_1',a_2')生成。为了说明这个结果,假定(a_1',a_2')$\in\mathcal{A}$,即 $a_1'=a_1-c\theta,a_2'=a_2+\theta$,这里 $0\leqslant\theta\leqslant q-1$。$y_1$ 和 y_2 有以下表示:

$$
\begin{aligned}
y_1 &= k_1+a_1r\\
&= k_1+(a'_1+c\theta)r\\
&= (k_1+rc\theta)+a'_1r,\\
y_2 &= k_2+a_2r\\
&= k_2+(a'_2-\theta)r\\
&= (k_2-r\theta)+a'_2r
\end{aligned}
$$

这里所有的计算都是在 $\mathbb{Z}_q$ 中。也就是说,四元组(γ,r,y_1,y_2)也可以由有序对(a_1',a_2')来产生,这时用随机选择 $k_1'=k_1+rc\theta$ 和 $k_2'=k_2-r\theta$ 来产生(相同的)γ。我们已经注意到 A 没有公开 k_1 和 k_2,因此四元组(γ,r,y_1,y_2)没有泄露 A 实际使用$\mathcal{A}$中哪个有序对作为他的秘密指数。

这个安全性证明是相当漂亮和微妙的,它对于回顾协议中与安全性证明相关的一些特征也许是有帮助的。其基本思想是让 A 选择两个秘密指数而不是一个。在集合$\mathcal{A}$中一共有 q 个有序对与 A 的秘密指数对(a_1,a_2)"等价"。导致最终矛盾结果的是知道$\mathcal{A}$中两个不同的对,就可以有效地计算出离散对数 c。当然,A 知道$\mathcal{A}$中一个有序对(a_1,a_2);证明了如果 O 能够假冒 A,那么 O 可以计算出(以大概率)$\mathcal{A}$中不同于(a_1,a_2)的另外一个有序对。这样 A 与 O 一起能找到$\mathcal{A}$中两个对,从而计算出 c,导致了矛盾。

Okamoto 签名算法:

系统的参数为 p、q、α_1、α_2 和 t,$q\mid(p-1)$,p 和 q 为大素数,p 至少为 512b 长,q 至少为 140b 长,t 是一个安全参数,t 至少为 20,α_1 和 α_2 都是 Z_p 中阶为 q 的元素。签名者 A 的公钥为 $v=\alpha_1^{-a_1}\alpha_2^{-a_2} \bmod p$,私钥是一对参数 $a_1,a_2,a_1,a_2\in Z_q$。H 是一个单向函数。

当 A 对消息 m 签名时,首先随机选择两个数 $r_1,r_2\in Z_q$,然后计算:

$$
\begin{aligned}
x &= \alpha_1^{r_1}\alpha_2^{r_2} \bmod p,\\
e &= H(x,m),\\
y_1 &= (r_1+ea_1) \bmod q,\\
y_2 &= (r_2+ea_2) \bmod q
\end{aligned}
$$

A 对消息 m 的签名是三重组(e,y_1,y_2)。

当接收者 B 收到签名(e,y_1,y_2)时,B 计算 $x=\alpha_1^{y_1}\alpha_2^{y_2}v^e \bmod p$,通过检验 $e=H(x,m)$ 是否成立来验证 A 的签名。

9.6　Guillou-Quisquater 身份识别协议

Guillou-Quisquater 身份识别协议[7]的安全性是基于 RSA 算法的安全性。

TA 选择两个大素数 p 和 q,形成 $n=pq$。保密 p 和 q,公开 n。TA 选择一个大素数 b,

b 用作安全参数和公开的 RSA 加密指数。一般假定 b 是一个 40b 长的素数。TA 选择一个签名算法和 Hash 函数。

TA 向证明者 A 颁布证书的协议如下。

(1) TA 建立 A 的身份并颁布一个标识串 ID(A)。

(2) A 秘密地选择一个整数 $u(0\leqslant u\leqslant n-1)$，$A$ 计算 $v=(u^{-1})^b \bmod n$，并将 v 发送给 TA。

(3) TA 对(ID(A)，v)签名，$s=\mathrm{Sig}_{TA}(\mathrm{ID}(A),v)$。并将证书 $C(A)=(\mathrm{ID}(A),v,s)$ 发送给 A。

证明者 A 向验证者 B 证明他的身份的协议，即 Guillou-Quisquater 身份识别协议如下。

(1) A 选择一个随机数 k，$0\leqslant k\leqslant n-1$，并计算 $\gamma=k^b \bmod n$。

(2) A 把他的证书 $C(A)=(\mathrm{ID}(A),v,s)$ 和 γ 发送给 B。

(3) B 通过检验 $\mathrm{Ver}_{TA}(\mathrm{ID}(A),v,s)$ 是否为真来验证 TA 的签名。

(4) B 选择一个随机数 $r(0\leqslant r\leqslant b-1)$，并将 r 发送给 A。

(5) A 计算 $y=ku^r \bmod n$，并将 y 发送给 B。

(6) B 通过检验 $\gamma\equiv v^r y^b \bmod n$ 是否成立来识别 A。

现在来说明 Guillou-Quisquater 身份识别协议是合理的和完备的。与通常一样，证明完备性是很简单的：

$$\begin{aligned} v^r y^b &\equiv (u^{-b})^r\,(ku^r)^b \pmod n \\ &\equiv u^{-br}k^b u^{br} \pmod n \\ &\equiv k^b \pmod n \\ &\equiv \gamma \pmod n \end{aligned}$$

现在来考虑合理性。将证明假设从 v 求出 u 是计算上不可行的，那么协议就是合理的。由于 v 是由 u 经过 RSA 加密得到的，因此这是一个可信的假设。

假定 O 知道 γ，他假冒 A 的成功概率是 $\varepsilon\geqslant\frac{2}{b}$。那么对于 γ，假定 O 能计算 y_1、y_2、r_1、r_2，满足 $r_1\neq r_2$，且

$$\gamma\equiv v^{r_1}y_1^b\equiv v^{r_2}y_2^b \pmod n$$

不失一般性，假定 $r_1>r_2$。那么有

$$v^{r_1-r_2}\equiv (y_2/y_1)^b \pmod n$$

由于 $0<r_1-r_2<b$ 且 b 是素数，$t=(r_1-r_2)^{-1} \bmod b$ 存在，O 能在多项式时间内通过欧几里德算法计算出 t。因此有

$$v^{(r_1-r_2)t}\equiv (y_2/y_1)^{bt} \pmod n$$

由于存在整数 ℓ，使得

$$(r_1-r_2)t=\ell b+1$$

因此

$$v^{\ell b+1}\equiv (y_2/y_1)^{bt} \pmod n$$

或等价地

$$v\equiv (y_2/y_1)^{bt}(v^{-1})^{\ell b} \pmod n$$

如果等式两边同时幂 $b^{-1} \bmod \phi(n)$ 运算，则

$$u^{-1} \equiv (y_2/y_1)^t (v^{-1})^\ell (\bmod n)$$

最后，等式两边同时进行模逆运算，计算出 u，即

$$u = (y_1/y_2)^t (v)^\ell (\bmod n)$$

O 可以用此公式在多项式时间内计算出 u。

Guillou-Quisquater 签名算法：

签名者 A 选择两个大素数 p 和 q，形成 $n=pq$。A 选择一个大素数 b 和一个 Hash 函数 H。A 秘密选择一个整数 u，$\gcd(u,n)=1$，将 u 作为他的私钥。A 的公钥为 $v=(u^{-1})^b \bmod n$。公开 n、b、v、H，保密 p、q 和 u。

A 对消息 m 的签名过程如下。

(1) 随机选择一个整数 k，$0 \leqslant k \leqslant n-1$，计算 $r=k^b \bmod n$。

(2) 计算 $e=H(m,r)$。

(3) 计算 $y=ku^e \bmod n$，A 对消息 m 的签名是数对 (e,y)。

接收者 B 验证签名的过程如下。

(1) 获得 A 的公钥 n、b、v。

(2) 计算 $r'=y^b v^e \bmod n$ 和 $e'=H(m,r')$。

(3) 验证是否有 $e=e'$，如果 $e=e'$，则 B 接收 A 的签名；否则拒绝。

9.7 GPS 身份识别协议

GPS 身份识别协议是欧洲 NEESIE 工程[8,9]的一个候选方案，该协议是一个交互式零知识识别协议，它具有以下特点：在一般任意模数离散对数问题（它等价于整数分解问题和计算素数模的离散对数问题）难解的假设下是可证明安全的，基于身份的短密钥，非常小的信息传输量和最小化的在线计算量。

设 k、l、A、B、S 是 5 个整数，$n=p\times q$ 是一个 kb 模数，p 和 q 是两个 $k/2$b 的素数，g 为 Z_n^* 中的一个元素，假定计算以 g 为底模 n 的离散对数是困难的。保密或销毁 p、q，公开 n。

从$[0,S)$中选取私钥 s，而公钥 I 则由 $I=g^{-s} \bmod n$ 确定。令 $\Phi=(B-1)(S-1)$，GPS 身份识别协议的一轮识别过程如下。

(1) 证明者从$[0,A)$中随机选取一个 r，计算“委托”$x=g^r \bmod n$ 并发送给验证者。

(2) 验证者从$[0,B)$中随机选取一个“挑战”c 发送给证明者。

(3) 证明者验证 $c\in[0,B)$，计算 $y=r+c\times s$，并将 y 发送给验证者。

(4) 验证者验证 $x=g^y I^c \bmod n$ 及 $y\in[0,A+\Phi)$是否成立。

重复上述过程 l 次。

GPS 识别协议与 Schnorr 识别协议相比，有以下几个不同点。

(1) 用一个合数代替了原来的素数 p 作为模。

(2) 没有了原来的参数 q，元素 g 的阶也不知道，而且 y 是在 Z 中计算的。

(3) 在区间$[0,A)$中选取随机值 r，使得私钥 s 隐藏于 $y=r+cs$ 之中。

(4) 证明者必须验证 c 在正确范围之内，验证者必须检查 y 值不能太大。

下面来证明 GPS 识别协议的安全性。首先给出将要用到的安全模型的精确描述，然后采用 Feige、Fiat 和 Shamir 的方法来证明 GPS 识别协议对主动敌手攻击的安全性，亦即要

证明完全性(Completeness)、合理性(Soundness)和零知识性。

9.7.1 安全模型

一个识别协议的目的是证明者要使验证者相信他自己的身份。证明者和验证者都可用概率多项式时间图灵机 PPTM 来模型化。这样的一个 PPTM($|n|$)具有一条特殊的通信带 ω,它的初始状态是布满随机均匀选择的比特。证明者和验证者还有其他可供读/写的通信带。

下面考虑如图 9.4 所示的交互识别环境。首先使用随机通信带 ω_{pp} 的一个随机化算法,对安全参数 k 产生公共参数 pp;然后使用随机通信带 ω_k 的一个概率算法,产生公私钥对(pk,sk)并发送私钥给证明者,公开相应的公钥;最后,识别就是在证明者和验证者之间进行一个交互式协议,最终所导致的结果就是验证者要么接受要么拒绝。

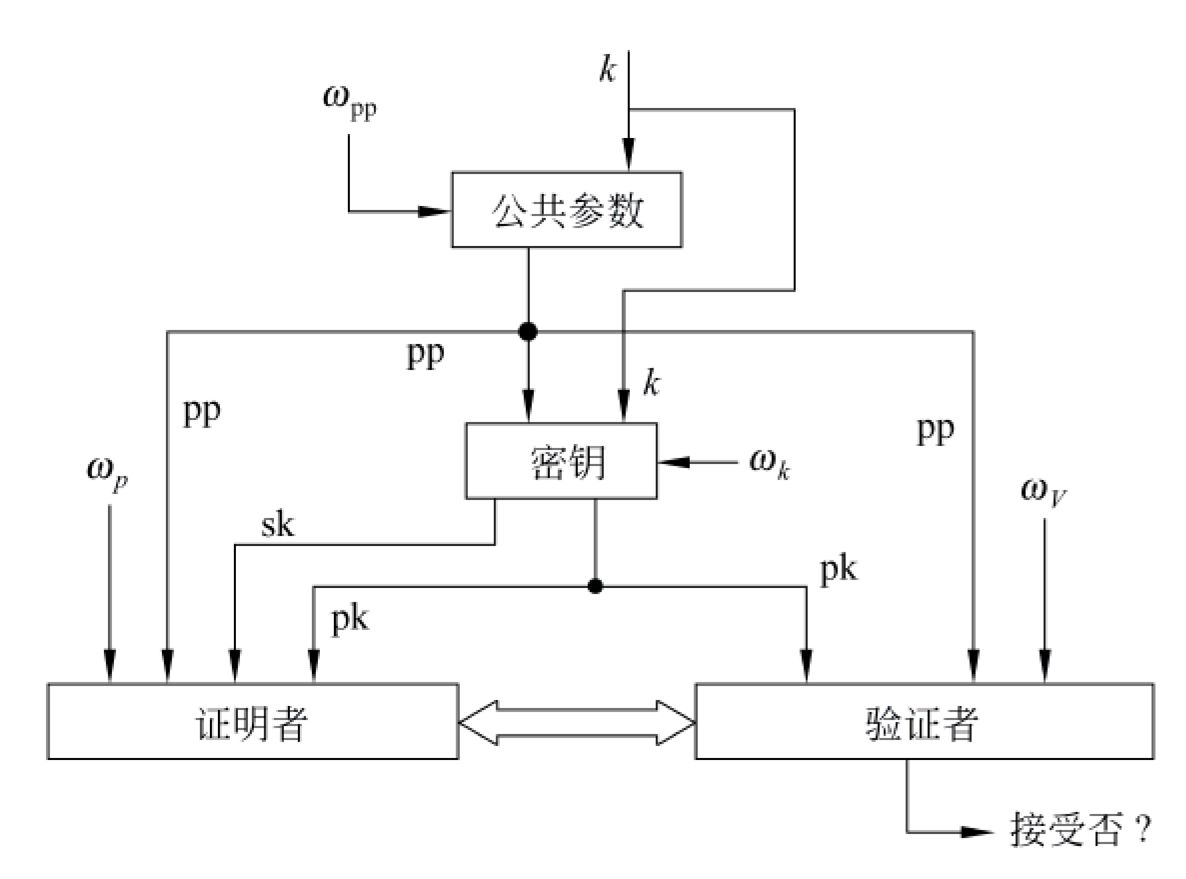


图 9.4 交互识别环境

在上述环境中,每一个人都是诚实的。现在考虑一下这种情形的一种变形:加入一个试图假冒证明者的攻击者。针对证明者的公钥,他试图使验证者接受他作为真实证明者。在这个模型中,不考虑对公共参数和密钥产生的攻击。因此,对攻击者而言,他只有两条途径来获得信息:首先,他可以被动地观察证明者和验证者之间常规认证的通信;其次,一个主动攻击者能够假扮成验证者或证明者。主动攻击和被动攻击之间的区别就在于主动攻击者会偏离诚实验证者遵守的协议,他这样做就是为了提取更多关于证明者私钥的信息。

只考虑不与证明者和验证者同时交互通信的攻击者。因此,一个这样的攻击者就可以看作两个概率多项式时间图灵机:第一个"攻击者(1)",他假冒验证者与证明者进行多项式次数的认证;第二个"攻击者(2)",他假扮成原来的证明者与验证者通信,参见图 9.5。注意:在这个安全模型中不考虑同时进行的攻击,也就是说,攻击者与证明者进行并发的认证或者重置攻击。更进一步,经典的中间人侵攻击并不能进行,因为把与证明者的交互和与验证者的交互区分开了。

现在在这个安全模型下来给出安全识别协议的定义。

称一个识别协议是安全的,如果对任何概率多项式时间攻击者,他被接受的概率都是可忽略的,即 $\forall d \in N, \exists k_0, \forall k > k_0$,有

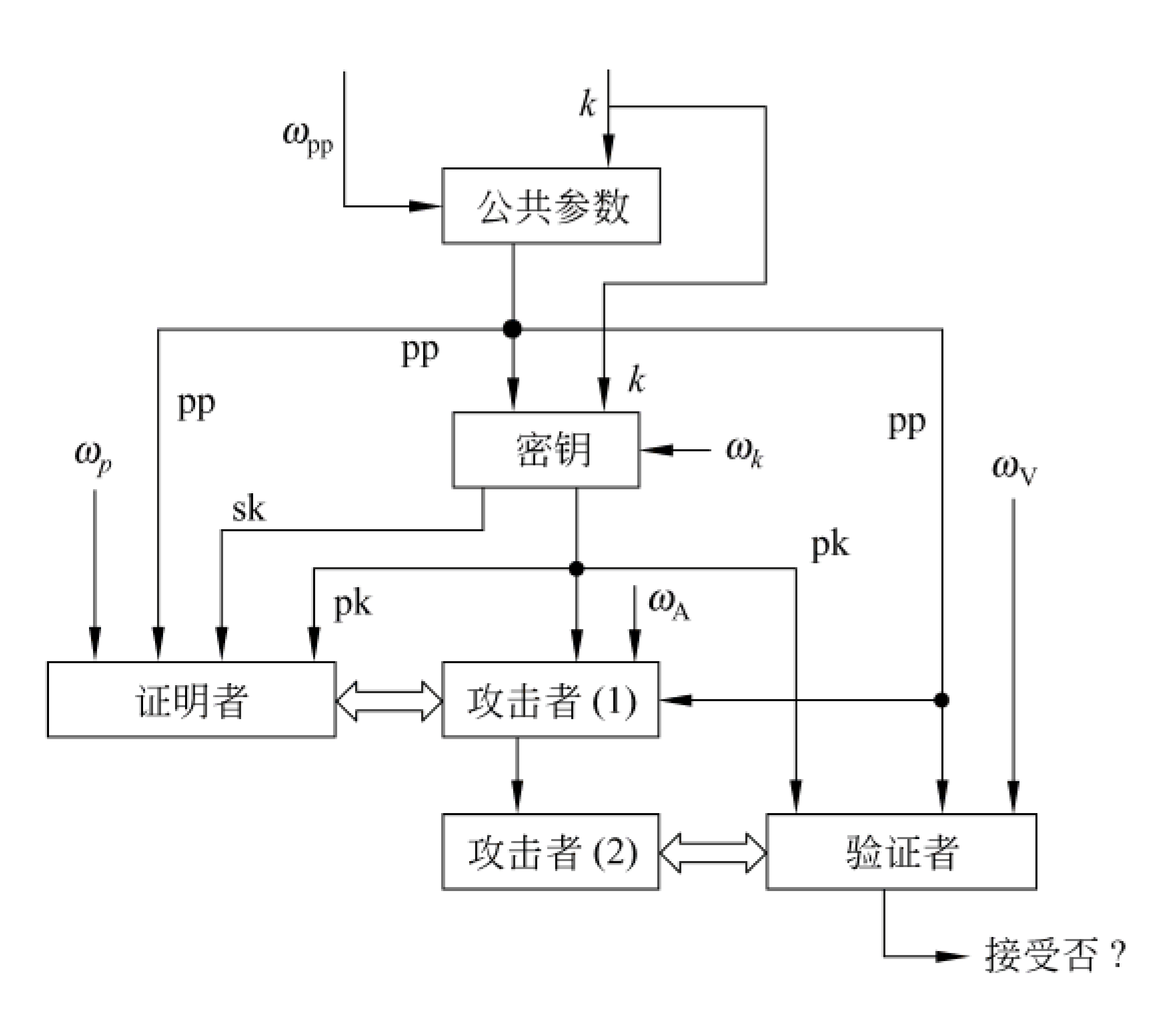


图 9.5　安全模型

$$\Pr_{\omega_{pp},\omega_k,\omega_p,\omega_V,\omega_A}\{\text{验证者接受攻击者}(2)\} < \frac{1}{k^d}$$

换言之，只有对足够大的安全参数 k，对随机选择的参数和密钥，任何多项式时间攻击者成功的概率都是可忽略的，识别协议才被认为是安全的。当然，这并不意味着某个攻击者不能针对某一特殊公钥来进行假冒（攻击者可以自己使用常规的密钥生成算法来产生一对公、私钥，对这样产生的公钥，攻击者只需遵守协议，就可以轻易被接受），这只是说攻击者不可能设计一个一般的假冒者。在渐近环境中，能够攻破某一特定密钥并不具有太大的意义。

9.7.2　GPS 的安全性证明

首先引进一些记号。对任何整数 x，用 $|x|$ 记 x 的二进制表示的比特数（$\lfloor \log_2(x) \rfloor + 1$）；定义函数 ζ：$\zeta(\text{true})=1$，$\zeta(\text{false})=0$；用概率多项式时间图灵机 PPTM(k)作为计算模型，它的运行时间是安全参数 k 的多项式量级；最后，用"Λ"表示逻辑"与"运算。

取安全参数 $k=|n|$，S、A、B、l 都是 k 的函数。为了简化记号，并不写明对 k 的依赖关系。当说 f 是可忽略的，意指对任何常数 d 和足够大的 k，有 $f(k)<1/k^d$。

为了证明安全性，首先来证明对诚实的证明者而言，他能够被正确地认证，即完全性(Completeness)。然后，零知识性表明一个攻击者与证明者的通信是可以模拟的。这表明图 9.5 中的证明者可以被一个不知道私钥的模拟器取代。由于与模拟器的通信过程和与真实的识别过程是不可区分的，所以攻击者察觉不到这个变化。因此，攻击者从模拟器得到的信息与从真实的证明者那儿得到的信息一样多。这样就可以推断，在协议执行过程中没有泄露任何关于私钥的信息。

最后证明，如果一个攻击者能以一个不可忽略的概率被接受，那么，基于此事实可构造一个算法在多项式时间内解决一个事先设定的困难问题，即合理性(Soundness)。在这个证明中，改动了一下密钥生成算法，但改动后的密钥分布与真正的密钥分布是不可区分的。因此推断，如果在模型中存在一个 PPTM($|n|$)攻击者，那么以 g 为底模 n 的离散对数就可以在 $|n|$ 的多项式量级时间内算出。由于假定这个问题对足够大的 n 和阶数足够大的底是困

难的，所以 GPS 识别协议对主动攻击敌手而言是安全的。

出于效率的原因，私钥在$[0,S)$内选择，并且没有模 g 的未知阶。因此，GPS 的安全性并没有归结为离散对数问题，准确地说是归结为所谓的短指数离散对数问题。

设 n 是一个模数，$g\in Z_n^*$，设 S 是小于 g 的阶的一个整数，短指数离散对数问题是指：给定 $y\in Z_n^*$，寻找 $x\in[0,S)$，满足 $y=g^x \bmod n$。

定理 9.4（完全性） 对于一个知道其公钥所对应的私钥的证明者和一个验证者而言，GPS 识别协议的执行总是成功的。

证明 在每一轮结束，验证者都得到 $x=g^r \bmod n$ 和 $y=r+cs$，其中后者对知道私钥 s 的证明者而言是很容易计算的。因此，

$$g^yI^c = g^{r+cs}g^{-cs} = g^r = x \bmod n$$

再者，

$$0\leqslant y = r+cs \leqslant (A-1)+(B-1)(S-1) < A+\Phi$$

合理性就是要证明：如果验证者以很大概率接受了一个证明者，那么，证明者必定知道 I 的以 g 为底模 n 的离散对数。直观地讲，在发送"委托"x 之后，如果证明者能以大于 $1/B$ 的概率正确响应 y，那么对两个不同的"挑战"c 和 c'，他一定能够正确响应 y 和 y'，满足 y，$y'<A+\Phi$，$g^yI^c=x=g^{y'}I^{c'} \bmod n$。设 $\sigma=y-y'$，$\tau=c'-c$，得到 $g^\sigma=I^\tau \bmod n$。下面的引理 9.1 正式地陈述了这些思想，其中暗含着 ε 依赖于 $k=|n|$。

引理 9.1 假设对某个 PPTM($|n|$)敌手，对某一公钥 I，他以 $\varepsilon>1/B^l$ 的概率被接受，那么存在一个算法，它以大于 $\frac{1}{6}\left(\frac{\varepsilon-1/B^l}{\varepsilon}\right)^2$ 的概率输出$(\sigma,\tau)\in(-A-\Phi,A+\Phi)\times(0,B)$，满足 $g^\sigma=I^\tau \bmod n$。该算法的平均运行时间$<2/\varepsilon t$，这里 t 是一次证明执行所需的平均运行时间。

证明 假设某个随机通信带为 ω 的 PPTM($|n|$)敌手 $\tilde{P}(\omega)$，对某一公钥 I，以 $\varepsilon=1/B^l+\varepsilon'$ 的概率被接受。在"挑战"为 $c_1,\cdots,c_l$ 的情况下，$\tilde{P}(\omega)$识别的结果（成功或失败）记为 $\mathrm{Succ}(\omega,c_1,\cdots,c_l)\in\{\mathrm{true},\mathrm{false}\}$。则

$$\Pr_{\omega,c_1,\cdots,c_l}\{\mathrm{Succ}(\omega,c_1,\cdots,c_l)\}\geqslant\varepsilon=1/B^l+\varepsilon'$$

构造以下算法。

(1) 选择一个随机带 ω 和一个 l 元数组 $c=(c_1,\cdots,c_l)$，$c_i\in[0,B)$ 直到 $\mathrm{Succ}(\omega,c)$ 成功。记试验次数为 u。

(2) 试验 u 次随机 l 元数组 $c'\neq c$ 直到 $\mathrm{Succ}(\omega,c')$ 成功；如果在 u 次试验后没有找到这样的 c'，那么算法失败。

(3) 设 j 是满足 $c_j\neq c_j'$ 的下标，用 y_j 和 y_j' 分别记 $\tilde{P}(\omega)$的相应正确响应。如果 $c_j>c_j'$，那么算法输出 $\sigma=y_j'-y_j$，$\tau=c_j-c_j'$；否则输出 $\sigma=y_j-y_j'$，$\tau=c_j'-c_j$。

如果算法不失败，那么证明者能够对两个不同的"挑战"c 和 c'，正确响应 y 和 y'，满足 y，$y'<A+\Phi$，$g^yI^c=x=g^{y'}I^{c'} \bmod n$。因此，$g^\sigma=I^\tau \bmod n$，$\sigma\in(-A-\Phi,A+\Phi)$，$\tau\in(0,B)$。

现在来分析一下上述算法的复杂度。由假设可知，$\tilde{P}(\omega)$成功的概率是 ε，因此第(1)步以同样的概率找到 ω 和 c；重复实验次数的期望值是 $1/\varepsilon$，试验次数 u 以 $\varepsilon(1-\varepsilon)^{N-1}$ 的概率取值 N。

设 Ω 是满足 $\Pr_c\{\mathrm{Succ}(\omega,c)\}\geqslant\varepsilon-\varepsilon'/2=1/B^l+\varepsilon'/2$ 的随机通信带 ω 的集合。在

$\mathrm{Succ}(\omega,c)=\mathrm{true}$ 的条件下，第(1)步找到的随机带 ω 属于 Ω 的条件概率的下界为

$$\begin{aligned}\Pr_{\omega,c}\{\omega\in\Omega\mid\mathrm{Succ}(\omega,c)\}&=1-\Pr_{\omega,c}\{\omega\notin\Omega\mid\mathrm{Succ}(\omega,c)\}\\&=1-\Pr_{\omega,c}\{\mathrm{Succ}(\omega,c)\mid\omega\notin\Omega\}\times\frac{\Pr_{\omega,c}\{\omega\notin\Omega\}}{\Pr_{\omega,c}\{\mathrm{Succ}(\omega,c)\}}\\&\geqslant 1-\left(\frac{1}{B^l}+\frac{\varepsilon'}{2}\right)\times 1/\varepsilon=\frac{\varepsilon'}{2\varepsilon}\end{aligned}$$

随机带 ω 以 $>\varepsilon'/2\varepsilon$ 的概率属于 Ω，这时由集合 Ω 的定义可知，能使算法成功的数组 $c'\neq c$ 的概率 $\geqslant\varepsilon'/2$，少于 N 次试验就得到这样一个 c' 的概率 $\geqslant 1-(1-\varepsilon'/2)^N$。

因此，得到这样一个 $\omega\in\Omega$ 和找到一个 c' 的概率就大于：

$$\begin{aligned}&\frac{\varepsilon'}{2\varepsilon}\sum_{N=1}^{+\infty}\varepsilon\times(1-\varepsilon)^{N-1}\left[1-\left(1-\frac{\varepsilon'}{2}\right)^N\right]\\&=\frac{\varepsilon'}{2}\left(\sum_{N=0}^{+\infty}(1-\varepsilon)^N-\left(1-\frac{\varepsilon'}{2}\right)\sum_{N=0}^{+\infty}\left[(1-\varepsilon)\left(1-\frac{\varepsilon'}{2}\right)\right]^N\right)\\&=\frac{\varepsilon'}{2}\left\{\frac{1}{\varepsilon}-\frac{1-\frac{\varepsilon'}{2}}{\varepsilon+\frac{\varepsilon'}{2}-\varepsilon\times\frac{\varepsilon'}{2}}\right\}\\&=\frac{\varepsilon'^2}{4\varepsilon^2(1+\varepsilon'/2\varepsilon-\varepsilon'/2)}>\frac{\varepsilon'^2}{6\varepsilon^2}\end{aligned}$$

因此，该算法以 $>\varepsilon'^2/6\varepsilon^2$ 的概率找到 (σ,τ)，$\widetilde{P}(\omega)$ 和验证者之间执行证明的总期望次数小于 $2/\varepsilon$。

如果 GPS 识别协议像 Schnorr 识别协议一样，已知 g 的阶，并且 g 的阶 $\mathrm{Ord}(g)$ 与区间 $(0,B)$ 中的任何整数互素，那么从 $g^\sigma=I^\tau \bmod n$ 恢复私钥 s 就很容易，只需解 $\sigma-s\tau=0 \bmod \mathrm{Ord}(g)$ 即可。但如果 g 的阶未知，即使假定一些更强的陈述，比如强 RSA 问题，也无法解这个等式。因此，GPS 识别协议并不是一个离散对数的知识证明。然而，在安全模型之下可证明它的安全性，只需假设计算以 g 为底模 n 的离散对数问题是困难的即可。现在先回顾一个概率方面众所周知的引理。

引理 9.2[10] 设 $A\subset X\cdot Y$，满足 $\Pr_{x,y}\{A(x,y)\}\geqslant\varepsilon$ 和 $\Omega=\{a\in X\mid\Pr_y\{A(a,y)\}\geqslant\varepsilon/2\}$，那么

$$\Pr_x\{x\in\Omega\}\geqslant\varepsilon/2$$

定理 9.5(合理性) 假设某个 PPTM($|n|$)敌手以不可忽略的概率被一个诚实验证者接受，$\log(|n|)=o(l\times|B|)$，l、B 都是 $|n|$ 的多项式量级，那么存在一个 PPTM($|n|$)能求解短指数离散对数问题。

证明 用 $\Gamma=\{I=g^s \bmod n, s\in[0,S)\}$ 记所有可能的公钥之集。设 $I_0\in\Gamma$，现在来描述一个能计算 I_0 的以 g 为底模 n 的离散对数的算法。

为了使证明简单，给出一个非均匀(Non-uniform)算法来计算离散对数，也就是一个依赖于攻击者成功概率的算法。然而并没真正使用这个概率值，而是用它来估计复杂度，所以可以很容易把这个算法转化成一个均匀(Uniform)算法，只需把下边描述的所有图灵机并行运行就可以了。

在安全模型中，一个攻击者成功的概率 π 是指

$$\pi = \Pr_{I \in \Gamma, \omega_A, \omega_V} \{对公钥 I, 敌手被接受\}$$

其中 ω_A 是敌手 PPTM($|n|$)的随机带，ω_V 是诚实验证者的随机带。概率 π 是在所有随机带和公钥上计算，因此，对某些公钥，这个成功概率值可能非常小。然而对不可忽略的密钥部分，成功概率不能“太小”。

设 Γ_0 是满足以下条件的公钥子集：

$$\Pr_{\omega_A, \omega_V} \{对公钥 I, 敌手被接受\} \geqslant \pi/2$$

引理 9.2 表明，公钥 I 属于这个子集的概率大于 $\pi/2$。对这样的密钥，引理 9.1 表明，存在一个 PPTM($|n|$)$A(I)$，在时间 $T<4/\pi\tau$ 内，以概率 $\varepsilon>(\pi/2-1/B^l)^2/(6(\pi/2)^2)$输出$(\sigma,\tau)\in(-A-\Phi,A+\Phi)\times(0,B)$满足 $g^\sigma=I^\tau \bmod n$。

因此，公钥 $g^s \bmod n$ 属于 Γ_0 并且 $A(I)$输出(σ,τ)满足 $g^\sigma=I^\tau \bmod n$ 的概率大于 $\pi/2\varepsilon$。根据 $A(I)$输出的(σ,τ)是否满足 $\sigma-s\tau=0$，可以分为以下两种情况进行讨论。

(1) 在大多数时间，$A(I)$输出(σ,τ)满足 $\sigma-s\tau=0$，立即得到所要找的离散对数$s=\sigma/\tau$。

(2) 如果 $A(I)$输出的(σ,τ)不满足 $\sigma-s\tau=0$，得到 g 的乘法阶的一个倍数，这个信息能用来解等式 $\sigma'-x\tau'=0 \bmod \mathrm{Ord}(g)$，从而由 $A(I)$的输出计算出离散对数。

情形 1：如果公钥 $g^s \bmod n$ 属于 Γ_0 并且 $A(I)$输出(σ,τ)满足 $\sigma-s\tau=0$ 的概率大于 $\pi\varepsilon/4$，则

$$\Pr_{s\in[0,S)}\{g^s \in \Gamma \wedge A(g^s)\ 输出(\sigma,\tau) \wedge \sigma - s\tau = 0\} \geqslant \pi\varepsilon/4$$

此时，可用下列算法计算 I_0 的离散对数。

(1) 选择 $r\in(-S,S)$。

(2) 计算 $I'=I_0\times g^r \bmod n$。

(3) 对输入 I'运行 A。

(4) 如果 $A(I')$输出(σ,τ)满足 $\tau|\sigma, \sigma/\tau+r\in[0,S)$ 和 $I_0=g^{\sigma/\tau+r} \bmod n$，输出 $\log_g I_0=\sigma/\tau+r$；否则返回(1)重新开始。

如果 $I_0\in\Gamma, r\in(-S,S), I'\in\Gamma$ 的概率是 1/2；更进一步，如果 $I'\in\Gamma$，它就是均匀分布的。因此，上述算法在大约 $8/\pi\varepsilon$ 次执行循环后可以找到 $\log_g I_0$，每次循环调用两次 A，因此该算法的期望运行时间是 $O(T/(\pi\varepsilon))$。

情形 2：现在考虑 A 没能立即输出私钥的情形。

$$\Pr_{s\in[0,S]}\{g^s \in \Gamma \wedge A(g^s)\ 输出(\sigma,\tau) \wedge \sigma - s\tau \neq 0\} \geqslant \pi\varepsilon/4$$

首先用下述算法计算 g 的乘法阶的一个倍数。

(1) 选择 $s_0\in[0,S)$。

(2) 计算 $I=g^{s_0} \bmod n$。

(3) 对输入 I 运行 A。

(4) 如果 A 没输出满足 $L_0=\sigma-s_0\tau\neq0$ 的(σ,τ)从(1)重新开始；否则输出 L_0。

经过一个期望运行时间 $O(T/(\pi\varepsilon))$，得到 L_0 满足

$$g^{L_0} = g^{\sigma-s_0\tau} = g^\sigma/I^\tau = 1 \bmod n$$

因此，L_0 是 g 的阶的一个小于 $A+\Phi$ 的倍数。更进一步，可以丢弃 L_0 小于 B 的非 g 之阶的因子。直观来说，使用 Pollard-rho 方法可以在 $O(\sqrt{B})$的时间内完成这件事。但如果需要

一个严格的证明，试除法测试可以在 $O(B)$ 时间内完成这件事。最后得到 L_0 满足 $g^{L_0}=1$ mod n，$L_0/\mathrm{Ord}(g)$不含有任何小于 B 的素因子。

现在，计算以 g 为底的离散对数。

(1) 选择 $r\in(-S,S)$。

(2) 计算 $I'=I_0\times g^r \bmod n$。

(3) 对输入 I'运行 A。

(4) 如果 A 不输出(σ,τ)，从步骤(1)重新开始；否则输出(σ,τ)。

得到 I'，(σ,τ)满足 $I'^{\tau}=g^{\sigma} \bmod n$。为了得到 I'的离散对数，解等式 $\sigma-\tau x=0 \bmod L_0$。由于 $B>\tau$，$L_0/\mathrm{Ord}(g)$不含有任何小于 B 的素因子，设 $d=\gcd(\tau,L_0)=\gcd(\tau,\mathrm{Ord}(g))$，等式$(\sigma/d-(\tau/d)x=0 \bmod L_0/d)$仅有一个根 $x_0=(\sigma/d)\times(\tau/d)^{-1} \bmod L_0/d$，所以 $x=x_0+iL_0/d \bmod L_0$，其中 $i\in[0,d)$。因此满足 $I'=g^x \bmod n$ 的解x可以在满足$d<\tau<B$ 的解中寻找。当然可以使用穷搜索方法，不过使用 Baby-step Giant-step 算法可以在时间 $O(\sqrt{B})$ 内完成这件事。

最后，在时间 $O(T/(\pi\varepsilon)+\sqrt{B})$内可得到满足 $I_0=g^{s_0} \bmod n$ 的 s_0。剩下的最后一个问题就是 s_0 有可能不在$[0,S)$范围之内。

在 $s\in[0,S)$的条件下，$A(g^s)$输出(σ,τ)并且上述算法正好输出 s 的概率大于$\pi\varepsilon/4$，在 $O(T/(\pi\varepsilon)+\sqrt{B})$内得到 $s_0\in[0,S)$；否则可以对输入 $I=g^s \bmod n$ 运行算法，并得到 s'满足 $g^{s'}=I=g^s \bmod n$ 而 $s'\neq s$。因此 $s'-s$ 和 L_0-s+s_0 都是 g 之阶的倍数，而且此二者之一小于 $L_0/2$。得到一个 L_0 的新值，然后可以重复上述步骤。随着 L_0 的减小，最终能够计算 I_0 在$[0,S)$范围之内的离散对数。因为初始 L_0 值的规模是$|A+\Phi|$，所以上述过程重复次数不超过$|A+\Phi|$。最后该算法的期望运行时间复杂度就是 $O(|A+\Phi|(T/(\pi\varepsilon)+\sqrt{B}))$。

如果$\pi(|n|)$不可忽略，那么存在整数 d 对无限多的$|n|$值，满足 $\pi(|n|)\geqslant 1/|n|^d$。更进一步，由于 $\log(|n|)=o(l|B|)$，对足够大的$|n|$有 $1/B^l<1/2|n|^d$。因此，由引理 9.1，PPTM$(|n|)A$ 成功的概率 $\varepsilon>1/6(\varepsilon-1/B^l/\varepsilon)^2>1/24$，它的平均运行时间 T 小于 $4/\pi\tau<4|n|^d\tau$。最后，如果 B 是多项式量级，可以得到一个计算以 g 为底，范围在$[0,S)$之内的离散对数的算法。

定理 9.6(零知识性) 如果 l、B 都是多项式量级的，并且 lSB/A 是可忽略的，那么 GPS 识别协议就是统计零知识的。

证明 描述一个证明者 P 和不诚实验证者 $\tilde{V}$ 之间通信的一个多项式时间模拟。假设不诚实验证者 $\tilde{V}$ 为了获得有关s 的信息而没有随机选择“挑战”。设在第 i 轮识别时 $\tilde{V}$ 从之前与 P 的交互中已获得的数据为 hist。此时证明者发送委托 x_i，$\tilde{V}$(有可能用到 hist 及 x_i)从随机带 $\omega_{\tilde{V}}$选择“挑战”$c_i(x_i,\mathrm{hist},\omega_{\tilde{V}})$。

下面是第 i 轮识别的一个模拟。

(1) 随机选择$\overline{c_i}\in[0,S)$和$\overline{y_i}\in[\Phi,A)=[(B-1)(S-1),A)$。

(2) 计算$\overline{x_i}=g^{\overline{y_i}}I^{\overline{c_i}} \bmod n$。

(3) 如果 $c_i(\overline{x_i},\mathrm{hist},\omega_{\tilde{V}})\neq\overline{c_i}$，那么回到(1)尝试另一对$(\overline{c_i},\overline{y_i})$；否则返回$(\overline{x_i},\overline{c_i},\overline{y_i})$。

上述循环的期望执行次数为 B。因此，l 轮上述模拟的复杂度为 $O(lB)$。下面来证明上述模拟产生的三元组的分布与真正识别产生的三元组的分布是统计不可区分的，也就是说：

$$\sum_1 = \sum_{\alpha,\beta,\gamma} \left| \Pr_{\omega_A,\omega_{\bar{V}}} \{(\alpha,\beta,\gamma) = (x,c,y)\} - \Pr_{\omega_M,\omega_{\bar{V}}} \{(\alpha,\beta,\gamma) = (\bar{x},\bar{c},\bar{y})\} \right|$$

是可忽略的。这意味着即使拥有无限的计算能力，这两个分布也不会被任何算法区分开，最多只能看到多项式个相同分布的三元组[11]。

设(α,β,γ)是一个固定的三元组，现在来估算一下在一轮证明和一轮模拟中得到这样一个三元组的概率。假设证明者是诚实的，也就是说遵守协议。则

$$\Pr_{\omega_P,\omega_{\bar{V}}} \{(x,c,y) = (\alpha,\beta,\gamma)\} = \Pr_{r\in[0,A),\omega_{\bar{V}}} \{\alpha = g^r \wedge \beta = c(\alpha,\text{hist},\omega_{\bar{V}}) \wedge \gamma = r + cs\}$$

$$= \sum_{r\in[0,A)} \frac{1}{A} \Pr_{\omega_{\bar{V}}}\{\alpha = g^\gamma/I^\beta \wedge \beta = c(\alpha,\text{hist},\omega_{\bar{V}}) \wedge r = \gamma - cs\}$$

$$= \frac{1}{A} \Pr_{\omega_{\bar{V}}}\{\alpha = g^\gamma/I^\beta \wedge \beta = c(\alpha,\text{hist},\omega_{\bar{V}}) \wedge \gamma - cs \in [0,A)\}$$

$$= \frac{1}{A} \times \delta(\alpha = g^\gamma/I^\beta) \times \Pr_{\omega_{\bar{V}}}\{\beta = c(\alpha,\text{hist},\omega_{\bar{V}})\} \times \delta(\gamma - cs \in [0,A))$$

其中δ定义为$\delta(\text{true})=1$，$\delta(\text{false})=0$。现在来考虑在模拟期间得到三元组(α,β,γ)的概率$\Pr_{\omega_M,\omega_{\bar{V}}}\{(\bar{x},\bar{c},\bar{y})=(\alpha,\beta,\gamma)\}$，这里$\omega_M$是模拟器的随机带。这个条件概率是

$$\Pr_{\bar{y}\in[\Phi,A),\bar{c}\in[0,B),\omega_{\bar{V}}} \{\alpha = g^{\bar{y}}/I^{\bar{c}} \wedge \beta = \bar{c} \wedge \gamma = \bar{y} \mid \bar{c} = c(g^{\bar{y}}/I^{\bar{c}},\text{hist},\omega_{\bar{V}})\}$$

应用 Bayes 法则，此概率可写成以下形式：

$$\frac{\Pr_{\bar{y}\in[\Phi,A),\bar{c}\in[0,B),\omega_{\bar{V}}} \{\alpha = g^{\bar{y}}/I^{\bar{c}} \wedge \beta = \bar{c} = c(\alpha,\text{hist},\omega_{\bar{V}}) \wedge \gamma = \bar{y}\}}{\Pr_{\bar{y}\in[\Phi,A),\bar{c}\in[0,B),\omega_{\bar{V}}} \{\bar{c} = c(g^{\bar{y}}/I^{\bar{c}},\text{hist},\omega_{\bar{V}})\}}$$

记$Q = \sum_{\bar{y}\in[\Phi,A),\bar{c}\in[0,B)} \Pr_{\omega_{\bar{V}}}\{\bar{c} = c((g^{\bar{y}}/I^{\bar{c}},\text{hist},\omega_{\bar{V}}))\}$，则前述分数的分母为

$$\Pr_{\bar{y}\in[\Phi,A),\bar{c}\in[0,B),\omega_{\bar{V}}} \left\{\bar{c} = c(g^{\bar{y}}/I^{\bar{c}},\text{hist},\omega_{\bar{V}}) = \frac{1}{(A-\Phi)B}Q\right\}$$

因此，

$$\Pr_{\omega_M,\omega_{\bar{V}}} \{(\bar{x},\bar{c},\bar{y}) = (\alpha,\beta,\gamma)\}$$

$$= \sum_{\bar{c}\in[0,B)} \frac{1}{B} \Pr_{\bar{y}\in[\Phi,A),\omega_{\bar{V}}} \{\alpha = g^{\bar{y}}/I^\beta \wedge \beta = \bar{c} = c(\alpha,\text{hist},\omega_{\bar{V}}) \wedge \gamma = \bar{y}\} \Big/ \frac{Q}{(A-\Phi)\times B}$$

$$= \Pr_{\bar{y}\in[\Phi,A),\omega_{\bar{V}}} \{\alpha = g^\gamma/I^\beta \wedge \beta = c(\alpha,\text{hist},\omega_{\bar{V}}) \wedge \gamma = \bar{y}\} \times \frac{A-\Phi}{Q}$$

$$= \sum_{\bar{y}\in[\Phi,A)} \frac{1}{A-\Phi} \times \Pr_{\omega_{\bar{V}}}\{\alpha = g^\gamma/I^\beta \wedge \beta = c(\alpha,\text{hist},\omega_{\bar{V}}) \wedge \gamma = \bar{y}\} \times \frac{A-\Phi}{Q}$$

$$= \frac{1}{Q} \times \delta(\alpha = g^\gamma/I^\beta) \times \Pr_{\omega_{\bar{V}}}\{\beta = c(\alpha,\text{hist},\omega_{\bar{V}})\} \times \delta(\gamma \in [\Phi,A))$$

现在来计算$\sum_1$。

$$\sum_1 = \sum_{\gamma\in[\Phi,A),\beta\in[0,B),\alpha=g^\gamma/I^\beta} \left| \frac{1}{A} \Pr_{\omega_{\bar{V}}}\{\beta = c(\alpha,\text{hist},\omega_{\bar{V}})\} - \frac{1}{Q} \Pr_{\omega_{\bar{V}}}\{\beta = c(\alpha,\text{hist},\omega_{\bar{V}})\} \right|$$

$$+ \sum_{\alpha,\beta,\gamma\notin[\Phi,A)} \Pr_{\omega_A,\omega_{\bar{V}}} \{(x,c,y) = (\alpha,\beta,\gamma)\}$$

$$= \left|\frac{1}{A} - \frac{1}{Q}\right| \times \left(1 - \sum_{\gamma\in[\Phi,A),\beta\in[0,B),\alpha=g^{\gamma}/I^{\beta}} \frac{1}{A} \Pr_{\omega_{\bar{V}}}\{\beta = c(\alpha,\text{hist},\omega_{\bar{V}})\}\right)$$

$$= \frac{|Q-A|}{A} + 1 - \frac{Q}{A}$$

为了完成证明，还需对 Q 的值进行估计：

$$Q = \sum_{\bar{y}\in[\Phi,A),\bar{c}\in[0,B)} \Pr_{\omega_{\bar{V}}}\{\bar{c} = c(g^{\bar{y}}/I^{\bar{c}},\text{hist},\omega_{\bar{V}})\}$$

$$= \sum_{\omega_{\bar{V}}} \Pr\{\omega_{\bar{V}}\} \sum_{\bar{y}\in[\Phi,A),\bar{c}\in[0,B)} \delta(\bar{c} = c(g^{\bar{y}}/I^{\bar{c}},\text{hist},\omega_{\bar{V}}))$$

设 f 是任意一个从 G 到$[0,B)$的函数。为了计算 Q，必须对满足 $f(g^{y}/I^{c})=c$ 的数对$(c,y)\in[0,B)\times[\Phi,A)$个数进行估计。首先注意到如果 $f(g^{y}/I^{c})=c$，那么对任意满足 $c+k\in[0,B)$的 k，有 $f(g^{y+kS}/I^{c+k})=c\neq c+k$。更进一步，如果对任意的 $k\in[-c,B-c)$，都有 $y+kS\in[\Phi,A)$，在 B 对$(c+k,y+kS)$中有且仅有一个是等式 $f(g^{y+kS}/I^{c+k})=c+k$ 的解。因此，可以把数对(c,y)的集合划分为 $A-\Phi-(B-1)S$ 个子集，其中每个子集恰含有一个解；也可划分成 $2(B-1)S$ 个子集，其中每个子集至多有一个解，参见图 9.6。因此，解的总数 N 满足

$$A-2\Phi \leqslant A-\Phi-(B-1)S \leqslant N \leqslant (A-\Phi-(B-1)S)+2(B-1)S \leqslant A$$

可以把这个结果应用到任意函数 f，特别地，应用到把 $c(g^{\bar{y}}/I^{\bar{c}},\text{hist},\omega_{\bar{V}})$和$(\bar{c},\bar{y})$联系起来的那个函数。因为 $\sum_{\omega_{\bar{V}}} \Pr\{\omega_{\bar{V}}\} = 1$，所以可推知 Q 在 $A-2\Phi$ 和 A 之间。最后，可以得到

$$\sum\nolimits_1 = \frac{|Q-A|}{A} + 1 - \frac{Q}{A} = 2(1-Q/A) \leqslant 4\Phi/A < 4SB/A$$

这就证明了如果SB/A是可忽略的，那么真正的分布与模拟的分布之间是统计不可区分的。为了完成证明，还需要说明当执行 l 轮模拟时，泄露的信息量增长得不太快，即有

$$\sum\nolimits_l = \sum_{(\alpha_i,\beta_i,\gamma_i),i\in[1,l]} \left| \Pr_{\omega_A,\omega_{\bar{V}}}\{(\alpha_i,\beta_i,\gamma_i) = (x_i,c_i,y_i)\} - \Pr_{\omega_M,\omega_{\bar{V}}}\{(\alpha_i,\beta_i,\gamma_i) = (\overline{x_i},\overline{c_i},\overline{y_i})\} \right|$$

其中 i 表示轮的标号。

已经证明这个量的上界当 $l=1$ 时，是$4SB/A$。现在用数学归纳法来证明这个量的上界对任意 l，是$4lSB/A$。用条件概率把前 $l-1$ 轮与最后一轮分开。记 $t_i=(\alpha_i,\beta_i,\gamma_i)$，$d_i=(x_i,c_i,y_i)$，$\overline{d_i}=(\overline{x_i},\overline{c_i},\overline{y_i})$。

$$\sum\nolimits_l = \sum_{t_i,i\in[1,l]} \left|\Pr\{t_i=d_i,\forall i\in[1,l]\} - \Pr\{t_i=\overline{d_i},\forall i\in[1,l]\}\right|$$

$$= \sum_{t_i,i\in[1,l-1]} \sum_{t_l} \left| \begin{array}{l} \Pr\{t_i=d_i,\forall i\in[1,l-1]\}\Pr\{t_l=d_l/t_i=d_i,\forall i\in[1,l-1]\} \\ -\Pr\{t_i=\overline{d_i},\forall i\in[1,l-1]\}\Pr\{t_l=\overline{d_l}/t_i=d_i,\forall i\in[1,l-1]\} \end{array} \right|$$

应用下列的常用加减项技巧，

$$|x\times y - z\times t| = |xy - xt + xt - zt| = |x(y-t) + t(x-z)|$$
$$\leqslant x|y-t| + t|x-z|$$

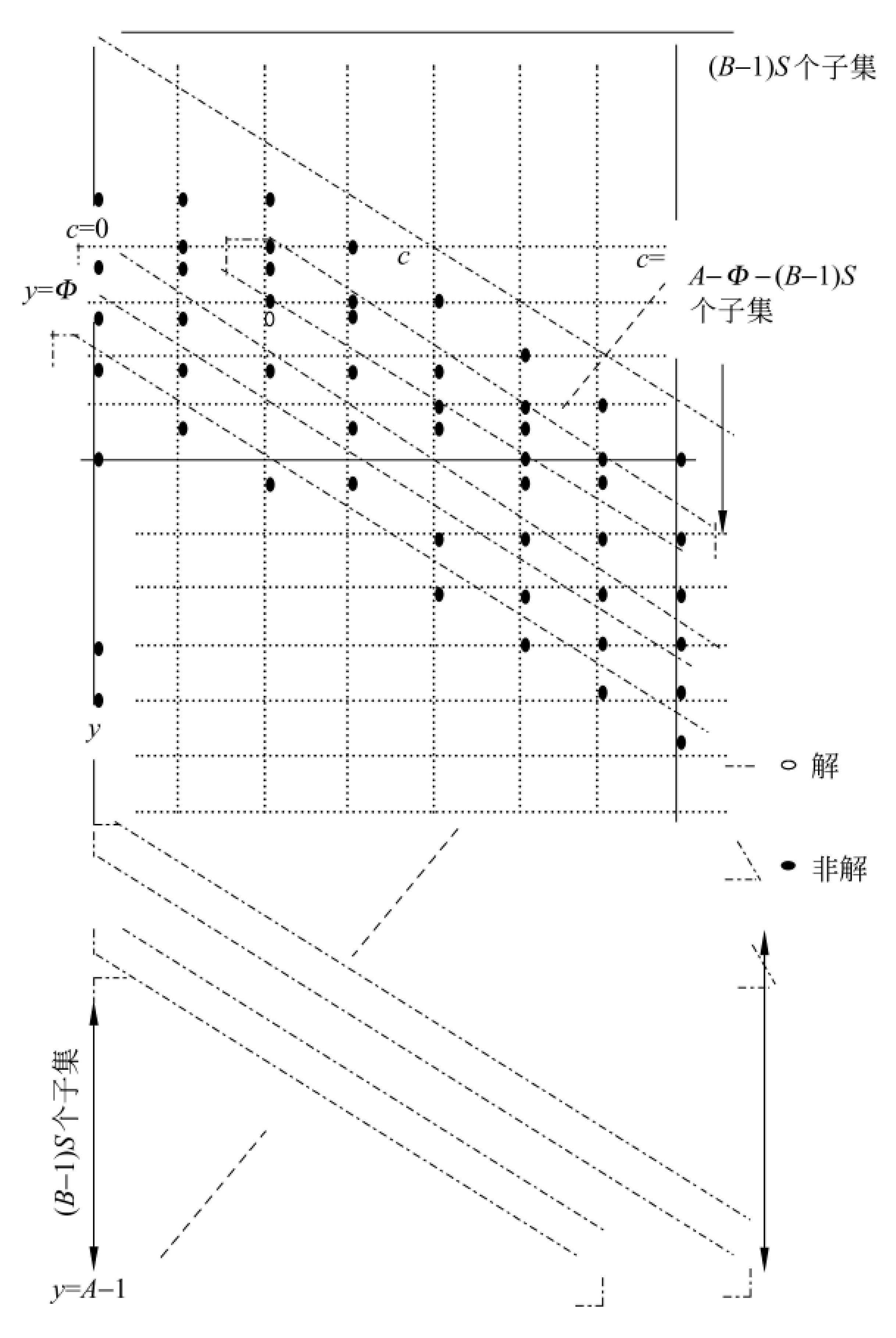


图 9.6 满足 $f(g^y/I^c)=c$ 的数对 $(c,y)\in[0,B)\times[\Phi,A)$ 个数估计

可以得到 $\sum\limits_{l}$ 的上界为

$$\begin{aligned}&\sum_{t_l}\left|\begin{matrix}\Pr\{t_l=d_l/\ t_i=d_i,\forall i\in[1,l-1]\}\\-\Pr\{t_l=\overline{d_l}/\ t_i=\overline{d_i},\forall i\in[1,l-1]\}\end{matrix}\right|\\&+\sum_{t_i,i\in[1,l-1]}\left|\begin{matrix}\Pr\{t_i=d_i,\forall i\in[1,l-1]\}\\-\Pr\{t_i=\overline{d_i},\forall i\in[1,l-1]\}\end{matrix}\right|\\&\leqslant\frac{SB}{A}+\frac{(l-1)SB}{A}=\frac{lSB}{A}。\end{aligned}$$

最后一步基于归纳假设和第一步的结果。

总之，如果 lSB/A 是可忽略的，l、B 是多项式量级的，那么 GPS 识别协议就是统计零知识的。

这个定理的一个结论是：一个诚实证明者与一个可能不诚实的验证者之间的通信只有在 B 是 $|n|$ 的多项式量级的情况下才有可能被模拟。这并不令人奇怪，因为 Schnoor 识别协议也有同样的问题。

定理 9.7（诚实验证者零知识性）　若 l 是多项式量级的并且 lSB/A 是可忽略的，则 GPS 识别协议是诚实验证者统计零知识的。

最后简要说明一下参数的选择，实际上这些参数之间的关系可从协议的证明过程中得出。

（1）n 的选择。在理论分析中，没有限制 n 的可能取值，只是把安全性归结为计算模 n 离散对数的难度。注意到如果知道 n 的因子分解 $\prod_i p_i^{e_i}$，那么由中国剩余定理，计算模 n 离散对数可归结为计算模上每一个 $p_i^{e_i}$ 的离散对数。因此，关于 n 的选择应遵循以下原则。

① n 可以是一个素数，如果素数模的规模大于 768b，那么素数模的离散对数问题可以认为是困难的；为了得到更高的安全性，可以在实际中取 $|n|=1024$b。

② n 也可以是一个合数。为了防止 n 被分解，$|n|$ 必须大于 1024b。进一步，n 的素因子必须具有相近的规模，而且因子个数不能太多。从理论角度来说，使用一个具有非常大的素数因子的模数可以把 GPS 方案的安全性同时基于离散对数问题和因子分解问题；从实用角度来讲，1024b 的模数就足以在未来数年里得到一个较高的安全水平。

因此在实际中，采用一个以下选择的 $k$bit 模数：用 Miller-Rabin 算法选取两个 $k/2$b 的素数 p、q，令 $n=pq$，保密或销毁 p、q，公开 n。

（2）g 的选择。基 g 的选择必须使得计算离散对数是困难的。因此，模数 n 一旦选定，g 就必须是一个阶数很大的元素。在许多实际应用中，选择 $g=2$ 并不减弱安全性，这样做的优点是，在使用“平方乘”算法计算模求幂时，开始的那些平方运算无需做模约简，而且乘 g 的运算就是移位操作，至多其后跟着一个减 n 的操作。

（3）S 的选择。S 的选择视计算离散对数算法的复杂度而定，比如 Pollard-Lambda 算法就可以在 $O(\sqrt{S})$ 内计算出离散对数，只需注意 $|S|$ 至少要大于 160b。

（4）B 的选择。B 的选择涉及敌手假冒成功的概率。期望的安全性依赖于实际应用，$|B|=32(l=1)$ 对许多识别协议来说就已经足够了，因为这保证了一个敌手不能以大于 2^{-32} 的概率来假冒一个用户（这正是 NESSIE 计划建议的安全水平）。

（5）A 的选择。当参数 S 和 k 选定之后，由于 A/BS 必须足够大以保证统计零知识特性，由此得到一个 A 的下界，比如可以取 $|A|\geqslant|S|+|B|+80$。

9.8　基于身份的识别协议

Shamir 于 1984 年提出了一类新型的密码方案[12]，这类方案能使网上的任何对用户无须交换私钥或公钥、无需保存密钥簿、无须使用第三方服务就可进行安全的通信和相互验证签名。在这类方案中，假定存在一个可信的密钥产生中心，该中心的主要作用是给每一个第一次入网的用户颁发一个个人化的 Smart 卡。嵌入在这个卡中的信息能使用户签名和加密他所发送的消息，并且能使用户用一个完全独立的方式解密和验证他所收到的信息。

Shamir 的基于身份的密码方案仍然是一类公钥密码方案，但它不是去直接生成一对随机的公钥和私钥，而是由用户选择他的名字和网络地址来作为公钥。当用户入网时，密钥产生中心首先对用户进行识别。如果接纳用户，密钥产生中心就为该用户以 Smart 卡的形式颁布一个私钥。于是，当一个网上用户想与另一个用户通信时，他只需知道对方的姓名和地

址就行了。这种方案与目前使用的邮政系统十分类似。

当用户 A 想给用户 B 发送一个消息 m 时，他就用自己的 Smart 卡中的私钥对消息 m 签名，用 B 的名字和网络地址作为密钥来加密签名和消息 m，然后将密文连同 A 的名字和地址发送给 B。当 B 收到消息时，他使用他的 Smart 卡中的私钥对消息进行解密，最后用发送者的名字和网络地址作为密钥对签名进行验证。

目前基于身份的算法和协议有很多。本节仅介绍 Shamir 的基于身份的数字签名算法[12]，以及 Guillou-Quisquater 的基于身份的识别协议和签名算法[13]。

1. Shamir 的基于身份的数字签名算法

该算法以下面的验证条件为基础：$s^e \equiv it^{f(t,m)} \pmod n$，其中 m 为消息，s，t 是签名，i 是用户的身份，n 是两个大素数的乘积，e 是一个大素数，f 是一个单向函数。

参数 n、e 和单向函数 f 由密钥产生中心生成，所有用户都有同样的 n、e 和 f。这些值可以公开，但 n 的分解只有密钥产生中心知道。不同用户之间的唯一差别就是与用户身份 i 对应的秘密密钥 g，这里 $g^e \equiv i \pmod n$。g 可由密钥产生中心很容易地计算出来，但是如果 RSA 体制是安全的，那么没有别的人能求出 i 关于模 n 的 e 次根。

签名者 A 对消息 m 的签名过程如下。

(1) 随机选择一个数 r，计算 $t = r^e \bmod n$ 和 $f(t,m)$。

(2) 计算 $s = gr^{f(t,m)} \bmod n$。

(3) A 对消息 m 的签名为 (t,s)。

接收者 B 通过下式来验证 (t,s) 是否为 m 的签名：

$$s^e \equiv it^{f(t,m)} \pmod n$$

2. Guillou-Quisquater 的基于身份的识别协议

Guillou-Quisquater 的基于身份的识别协议是由 Guillou-Quisquater 身份识别协议转化而来的。在这个协议中，TA 为用户 A 颁布一个数 u，u 是由 TA 用加密指数 a 计算的 RSA 密文，该数是 A 的身份 ID 串的一个函数，而无须为用户 A 颁发证书。参数的选择与 Guillou-Quisquater 身份识别协议的一样，这里 $ab \equiv 1 \pmod{\varphi(n)}$。

TA 为 A 颁布值 u 的过程如下。

(1) TA 建立 A 的身份并颁布一个标识串 ID(A)。

(2) TA 计算 $u = (H(\mathrm{ID}(A))^{-1})^a \bmod n$，并将 u 发送给 A。

其中 $H(\cdot)$ 是一个公开的 Hash 函数。

Guillou-Quisquater 的基于身份的身份识别协议的识别过程如下。

(1) A 随机选择一个数 k，$0 \leqslant k \leqslant n-1$，并计算 $\gamma = k^b \bmod n$。

(2) A 把 ID(A) 和 γ 发送给 B。

(3) B 计算 $v = H(\mathrm{ID}(A))$。

(4) B 选择一个随机数 r，$0 \leqslant r \leqslant b-1$，并将 r 发送给 A。

(5) A 计算 $y = ku^r \bmod n$，并将 y 发送给 B。

(6) B 通过检验 $\gamma = v^r y^b \bmod n$ 是否成立来识别 A。

v 是由公共 Hash 函数 H 对 A 的 ID 串作用而计算出的。为了执行身份识别协议，A 需要知道 u 值，u 只能由 TA 计算(假定 RSA 密码算法是安全的)。由于敌手不知道 u 值，

因而他无法假冒 A 向 B 证实身份。

3. Guillou-Quisquater 的基于身份的签名算法

签名者 A 选择两个大素数 p 和 q，形成 $n=pq$，并选择一个整数 e，使得 $\gcd(e,\varphi(n))=1$。选择一个整数 J_A，$1<J_A<n$，$\gcd(J_A,n)=1$，将 J_A 用作 A 的身份。J_A 的二进制表示能被用来反映 A 的某些个人信息，诸如姓名、地址、驾驶执照号等。A 按下列步骤计算 $a=(J_A^{-1})^{1/e} \bmod n$。

(1) 计算 $J_A^{-1} \bmod n$。

(2) 计算 $d_1=e^{-1} \bmod (p-1)$，$d_2=e^{-1} \bmod (q-1)$。

(3) 计算 $a_1=(J_A^{-1})^{d_1} \bmod p$，$a_2=(J_A^{-1})^{d_2} \bmod q$。

(4) 由中国剩余定理找到同余式方程组 $a\equiv a_1 (\bmod p)$，$a\equiv a_2 (\bmod q)$的一个解 a。

A 公开 n、e、J_A，保密 a，另外还选择并公开一个 Hash 函数 H。

签名者 A 对消息 m 的签名过程如下。

(1) 随机选择一个整数 k，计算 $r=k^e \bmod n$。

(2) 计算 $l=H(m,r)$。

(3) 计算 $s=ka^l \bmod n$。

(4) A 对 m 的签名是对 (l,s)。

接收者 B 验证签名的过程如下。

(1) 获得 A 的公钥 n、e、J_A。

(2) 计算 $u=s^eJ_A^l \bmod n$ 和 $l'=H(m,u)$。

(3) 验证是否有 $l=l'$，如果 $l=l'$，则 B 接收 A 的签名；否则拒绝。

9.9　小结

关于身份识别协议安全模型的研究可参阅文献[14]～[16]，文献[17]也做了一些介绍。将身份识别协议用于计算机网络中的研究最早是在 20 世纪 70 年代提出的[18]。本章写作过程中主要参考了文献[17]中的第 9 章。

Schnorr 身份识别协议和 Guillou-Quisquater 身份识别协议，在合理计算假设下的安全性证明可参阅文献[19]。Feige-Fiat-Shamir 身份识别协议[3]也是一个广泛使用的方案，利用零知识技术可以证明该方案的安全性。由身份识别协议构造签名算法的方法是由 Fiat 和 Shamir 提出的[2]，他们也提出了一个基于身份的身份识别协议。另外，我们在这一领域也取得了一些研究成果，感兴趣的读者可参阅文献[20]。

关于身份识别协议综述可参阅文献[21]和[22]。

参 考 文 献

[1] Blake-Wilson S, Menezes A J. Entity authentication authenticated key transport protocols employing asymmetric techniques. Lecture Notes in Computer Science, 1361 (1998), 137 ～ 158. (Fifth International Workshop on Security Protocols.)

[2] Fiat A, Shamir A. How to prove yourself: Practical solutions to identification and signature

problems, Advances in Cryptology-Crypto'86, springer-Verlag, 1987, 186～194.

[3] Feige U, Fiat A, Shamir A. Zero knowledge proofs of identity, proceedings of STOC, 1987, 210～219.

[4] Chase M, Anna Lysyankaya. On signatures of knowledge. In advances in Cryptology-Crypto2006.

[5] Schnorr C P. Efficient signature generation by smart cards. Journal of Cryptology, 4 (1991), 161～174.

[6] Okamoto T. Provably secure and practical identification schemes and corresponding signature Scheme, Advances in Cryptology-Crypto'92, Springer-Verlog, 1993, 31～53.

[7] Guillou L C, Quisquater J J. A practical zero-knownledge protocol fitted to security microprocessor minimizing both transmission and memory, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1989, 123～128.

[8] 冯登国,林东岱,吴文玲.欧洲信息安全算法工程.北京：科学出版社,2003.

[9] Girault M, Poupard G, Stern J. GPS, an asymmetric identification scheme for on the fly authentication of low cost smart cards, http://www.cosic.esat.kuleuven.ac.be/nessie.

[10] Pointcheval D, Stern J. Security Arguments for Digital Signatues and Blind Signatures, Journal of Cryptology, 2000, 13(3).

[11] Maurer U M, Yacobi Y. Non-interactive public-key Cryptography. In: Eurocrypt'91, LNCS547. Springer-Verlag, 1992, 498～507.

[12] Shamir A. Identity-based Cryptosystems and signature Schemes, Advances in Cryptology-Crypto'84, Springer-Verlag, 1985, 47～53.

[13] Guillou L C, Quisquater J J. A paradoxical Identity-based signature scheme resulting from zero-knownledge, Advances in Cryptology-Crypto'88, Springer-Verlag, 1990, 216～231.

[14] Diffie W, Van Oorschot P C, Wiener M J. Authentication and authenticated key exchanges. Designs, Codes and Cryptography, 2 (1992), 107～125.

[15] Bellare M, Rogaway P. Entity authentication and key distribution. Lecture Notes in Computer Science, 773 (1994), 232～249. (CRYPTO'93.)

[16] Blake-Wilson S, Menezes A J. Entity authentication authenticated key transport protocols employing asymmetric techniques. Lecture Notes in Computer Science, 1361 (1998), 137 ～ 158. (Fifth International Workshop on Security Protocols.)

[17] Stinson D R. Cryptography—Theory and Practice(Third Edition), CRC Press.(密码学原理与实践.冯登国 等译.北京：电子工业出版社,2009.)

[18] Needham R M, Schroeder M D. Using encryption for authentication in large networks of computers. Communications of the ACM, 21(1978), 993～999.

[19] Bellare M, Palacio A. GQ and Schnorr identification schemes: proofs of security against impersonation under active and concurrent attacks. Lecture Notes in Computer Science, 2442 (2002), 162～177(CRYPTO 2002).

[20] Xu J, Zhu W T. Dengguo Feng: An improved smart card based password authentication scheme with provable security. Computer Standards & Interfaces 31(4): 723～728 (2009).

[21] Burmester M, Desmedt Y, Beth T. Efficient Zero-knowledge identification schemes for smart cards. The Computer Journal, 35(1992), 21～29.

[22] DE Waleffe D, Quisquater J J. BETTER login protocols for computer networks. Lecture Notes in Computer Science, 741(1993), 50～70. (Computer Security and Industrial Cryptography, State of the Art and Evolution, 1991)

第10章　密钥交换协议

在Kerckhoff假设下，一个密码系统的安全性取决于对密钥的保护，而不是对系统或硬件自身的保护。密钥的保密和安全管理在信息系统安全中是极为重要的。密钥管理包括密钥的产生、存储、装入、分发、保护、销毁及保密等内容，其中分发和存储可能是最棘手的问题。密钥管理不仅影响系统的安全性，而且涉及系统的可靠性、有效性和经济性。当然，密钥管理过程中也不可能避免物理上、人事上、法规上等一些问题。本章主要介绍密钥管理中的密钥分发问题。

我们已经知道，相对于对称密码算法来说，公钥密码算法的优势在于无需交换密钥就可以建立安全信道。但遗憾的是，大多数公钥密码算法(如RSA)都比对称密码算法(如AES)速度慢。因此，在实际应用中对称密码算法通常用于加密“长”消息，这样就仍然需要解决密钥建立问题。

在密钥交换协议中，假定环境是一个拥有n个用户的不安全的网络，这些用户的名称唯一，有时也称为参与者、参与方等，他们的最终目标是获得一个共享秘密。在某些协议中，有可能还需使用一个可信权威(记为TA)，它负责诸如验证用户身份、颁发用户证书、为用户选择并传送密钥等事宜，根据不同情况，这个可信权威分别称之为可信第三方、认证服务器、密钥分发中心、密钥交换中心等；还有可能存在未授权的第三方，在不同的环境中可以有不同的名称，如敌手、入侵者、攻击者、窃听者和假冒者。

密钥交换协议(也称密钥建立协议)大致可分为以下两种。

(1) 密钥分配协议。密钥分配协议是指协议的一个参与方可以建立或者获得一个秘密值，并将此秘密值安全地分发给协议的其他参与方。

(2) 密钥协商协议。密钥协商协议是指协议的两个或者多个参与方共同提供信息，推导出一个共享密钥，在多数情况下要求参与协议的任何一方不能预先确定这个共享密钥。

密钥交换协议最终使得协议的参与者产生一个共享秘密，该秘密通常用来推导一个会话密钥。在理想状态下，会话密钥是一个临时秘密，即会话密钥的有效期会被限制在一小段时间内，如一次单独的通信连接或者会话，当会话结束时就把该会话密钥丢弃。会话密钥是非常有用的，这是因为以下几点原因。

(1) 会话密钥限制了敌手可以得到使用一个特定密钥加密的密文的数量，因为会话密钥是定期变化的。

(2) 只要协议设计良好，它们可以限制密钥泄露事件带来的暴露问题，可以用于泄露的可能性较高的“危险”环境中。

(3) 使用会话密钥常常能减少每个用户需要安全存储的长期信息的数量，因为只有在需要的时候一对用户才会生成会话密钥。

(4) 在不同的通信会话或者应用过程中建立的密钥相互独立。

在实际应用中，为了实现某个具体目标而要求密钥交换协议满足一些特殊的安全性质。下面是一些常见的安全性质。

(1) 已知会话密钥安全性。在某些密钥交换协议中，即使攻击者知道某些使用过的会话密钥，密钥交换协议仍然可以使得协议参与者使用该协议建立新的会话密钥。

(2) 前向安全性。在某些密钥交换协议中，某个时段使用的秘密信息暴露，并不能暴露以前建立的会话密钥。

(3) 不能使用泄露的身份信息冒充其他合法用户。在某些需要密钥确认(密钥确认是指使得一个参与者能够确认另一个参与者实际已经拥有某一个特定的密钥)的密钥交换协议中，如果某个攻击者得到其中一个用户 A 的秘密信息，他不能使用该信息冒充另一个合法用户 B 来与其他参与者完成密钥交换协议。

(4) 未知会话密钥共享。在某些密钥交换协议中，一个用户 A 不能使得他与一个用户 C 建立了会话密钥，而用户 C 却认为他是与用户 B 建立了会话密钥。

(5) 密钥控制。在某些密钥交换协议中，协议要求任何一个协议参与者都不能控制最终产生的会话密钥，也不能使得最终产生的会话密钥一定出现在某个预先设置的小范围内。

(6) 密钥的新鲜性。在某些密钥交换协议中，要求协议的参与者可以检验最终得到的会话密钥是一个新的密钥，而不是以前使用过的密钥。

10.1 密钥分配协议

密钥分配协议可以基于对称密码算法，也可以基于公钥密码算法来实现。本节将从两个方面介绍一些基本的、典型的密钥分配协议，其中一些可能是不安全的。值得一提的是，可以基于量子物理实现密钥分配，最典型的量子密钥分配协议是 Charles Bennett 和 Gilles Brassard 于 1984 年提出的量子密码方案，通常称为 BB84 方案。

10.1.1 基于对称密码算法的密钥分配协议

1. 无可信权威的密钥分配协议

在使用对称密码算法实现密钥分配协议时，如果系统中不存在认证服务器，可能需要协议的参与者一开始就共享一个长期秘密信息，称为长期密钥。基于对称密码算法的点对点密钥分配协议[1]利用了一个参与双方预先共享的对称密钥 K。这个对称密钥可以通过一个安全信道分发，也可以使用一个密钥预分配协议得到。

协议 10.1 简单的点对点密钥分配协议(基于对称密码算法 E)。

参数设置：

用户 A 和 B 预先共享一个对称密钥 K，并选定一个对称密码算法 E。

具体步骤：

(1) A 选择会话密钥 k，然后用共享的对称密钥 K 进行加密，即计算 $E_K(k)$ 并将其发送给 B。

(2) B 将 $E_K(k)$ 解密后得到会话密钥 k。

点对点密钥分配协议不提供前向安全性，一旦对称密钥 K 泄露，协议就完全失败了。

协议 10.1 使用了对称密码算法 E，这个密钥分配协议也可以通过使用单向函数来实现。

协议 10.2 简单的点对点密钥分配协议(基于单向函数 H)。

参数设置：

用户 A 和 B 预先共享一个长期密钥 K，并选定一个单向函数 H。

具体步骤：

(1) A 选择一个随机数 r，然后将 r 发送给 B。

(2) A 和 B 的会话密钥可以通过计算一个以 r 为变量的单向函数来实现，如计算 $k=H(K,r)$。

使用对称密码算法实现密钥分配协议的另一个实例是 Andrew RPC 协议[2]。这个协议有两个相互独立的组成部分：在前半部分中，两个用户通过共享的密钥 K 完成一次握手；在后半部分中，一个用户选择会话密钥，并将其发送给另一个用户。

协议 10.3 Andrew RPC 协议。

参数设置：

用户 A 和 B 预先共享一个长期密钥 K，并选定一个对称密码算法 E。

具体步骤：

(1) A 选择一个随机数 r_A，将其用长期密钥 K 加密，即计算 $E_K(r_A)$ 并将其发送给 B。

(2) B 解密 $E_K(r_A)$ 得到 r_A，将其数值加 1 后，连同自己选择的随机数 r_B 一起用 K 加密，即计算 $E_K(r_A+1,r_B)$ 并将其发送给 A。

(3) A 解密 $E_K(r_A+1,r_B)$ 得到 r_B，将其数值加 1，用 K 加密，即计算 $E_K(r_B+1)$ 并将其发送给 B。

(4) B 选取一个会话密钥 k 和一个新的随机数 r'_B，将它们用 K 加密，即计算 $E_K(k,r'_B)$ 并将其发送给 A。

协议 10.3 存在一个明显缺陷，即 A 并不能确认他得到的会话密钥是否是新鲜的。而且，协议的前三步与第(4)步相互独立，容易受到攻击。例如，攻击者可以将第(4)步中 B 发送给 A 的消息用前面的某个消息替换。

在使用对称密码算法实现密钥分配协议时，如果系统中不存在服务器，也可以不需要协议的参与者共享一个长期秘密信息，如 Shamir 无密钥协议[3]。Shamir 无密钥协议是一种只使用对称密码算法的密钥分配协议，它在既不要求共享密钥也不要求公钥的情况下，允许在公开信道上建立密钥。这个协议只能抵抗被动攻击，并且不能提供认证。

协议 10.4 Shamir 无密钥协议。

参数设置：

(1) 选择并公布一个公用素数 p，使得以 p 为模的离散对数是计算上不可行的。

(2) 用户 A 和 B 分别随机选择秘密数 a 和 b，其中 $1\leqslant a, b\leqslant p-2$，均与 $p-1$ 互素。然后分别计算 $c=a^{-1} \bmod (p-1)$ 和 $d=b^{-1} \bmod (p-1)$。

具体步骤：

(1) A 随机选择一个密钥 k，计算 $k^a \bmod p$ 并将其发送给 B。

(2) B 收到消息后，计算 $(k^a)^b \bmod p$，并将其发送给 A。

(3) A 将收到的值进行 c 次模 p 幂指数运算，从而“消除”以前的指数运算，得到 $k^b \bmod p$，并将其发送给 B。

(4) B 将收到的值进行 d 次模 p 幂指数运算，从而“消除”以前的指数运算，得到会话密钥 k。

2. 具有可信权威的密钥分配协议

在使用对称密码算法实现密钥分配协议时，系统中也可以存在认证服务器。此时，协议包含两个通信参与方 A 和 B，以及一个可信服务器 TA，假定 A、B 与服务器 TA 分别共享一个预先设定的长期密钥 K_A、K_B。

在有服务器的、使用对称密码算法实现的密钥分配协议中，Needham-Schroeder 共享密钥协议[4]是比较有名的一个。自从 1978 年提出以来，Needham-Schroeder 协议是许多基于服务器的认证和密钥分配协议的基础。

协议 10.5 Needham-Schroeder 共享密钥协议。

参数设置：

用户 A 和服务器 TA 的共享密钥是 K_A，用户 B 和服务器 TA 的共享密钥是 K_B。A、B 和 TA 所使用的对称密码算法的加密变换和解密变换分别记为 E 和 D。ID(A)和 ID(B)分别是 A 和 B 的标识符。

具体步骤：

(1) A 选择随机数 r_A，并将 ID(A)、ID(B)和 r_A 发送给 TA。

(2) TA 随机选择一个会话密钥 K，然后计算 $t_B=E_{K_B}(K \parallel \mathrm{ID}(A))$（称之为给 B 的票据）和 $y_1=E_{K_A}(r_A \parallel \mathrm{ID}(B) \parallel K \parallel t_B)$，并将 y_1 发送给 A。

(3) A 使用他的密钥 K_A 解密 y_1，得到 K 和 t_B。然后 A 将 t_B 发送给 B。

(4) B 使用他的密钥 K_B 解密 t_B，得到 K。然后 B 选择一个随机数 r_B 并计算 $y_2=E_K(r_B)$。B 将 y_2 发送给 A。

(5) A 使用会话密钥 K 解密 y_2 得到 r_B。然后 A 计算 $y_3=E_K(r_B-1)$并将 y_3 发送给 B。

在协议 10.5 中还需要一些有效性检验，这里的有效性检验是指验证解密的数据具有正确的格式，并且包含了预期的信息。注意到协议 10.5 没有使用消息认证码。这些有效性检验过程如下。

(1) 当 A 解密 y_1 时，他检验明文 $D_{K_A}(y_1)$是否具有以下格式：

$$D_{K_A}(y_1) = r_A \parallel \mathrm{ID}(B) \parallel K \parallel t_B\text{，对于某个 } K \text{ 和 } t_B$$

(2) 当 B 解密 y_3 时，他检查明文

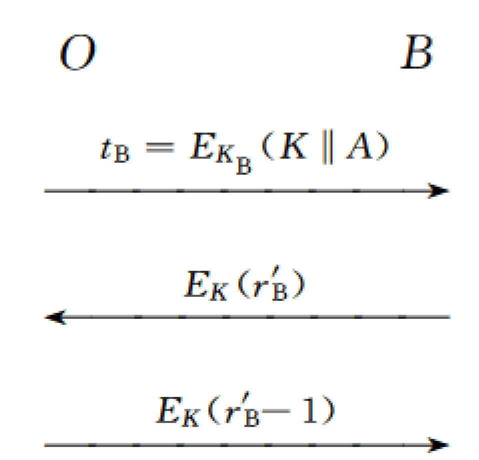
$$D_K(y_3) = r_B - 1$$

如果这一条件成立，则 B“接受”；否则 B“拒绝”。

注：有时用“A”表示“ID(A)”，用“B”表示“ID(B)”。

协议 10.5 的密钥确认是由 A 利用新的会话密钥加密挑战 r_B-1 来实现的，目的是为了使 B 相信 A 确实拥有了会话密钥 K。

1981 年，Denning 和 Sacco 发现了针对协议 10.5 的一种重放攻击[5]，现在来描述一下这种攻击。假设敌手 O 记录了在 A 和 B 之间进行的一次协议 10.5 的会话，记为 S，并以某种方式得到了会话 S 的会话密钥 K。这种攻击模型称为已知会话密钥攻击。然后 O 发起一次新的与 B 之间进行的协议 10.5 的会话，记为 S'，从会话 S' 的第 3 个流程开始，发送之前使用过的票据 t_B 给 B：



注意到,当 B 回复 $E_K(r'_B)$之后,O 能够使用已知的密钥 K 进行解密,然后减去 1,并对结果进行加密。在会话 S'的最后一个流程中,$E_K(r'_B-1)$被发送给 B。B 解密这条消息并"接受"该会话。

下面分析一下这个攻击的后果。在 O 与 B 进行的会话 S'的最后,B 认为他产生了一个"新"的会话密钥 K,并且 K 是与 A 共享的,这是因为在票据 t_B 中出现的是ID(A)。O 知道这个密钥,但是 A 未必知道,因为在与 B 进行的前一个会话 S 结束之后,A 可能已经扔掉了密钥 K。所以,在这个攻击中 B 从两个方面被欺骗了。

(1) B 所期望的对等方并不知道在会话 S'中分配的密钥 K。

(2) 会话 S'的密钥被除了 B 所期望的对等方之外的其他人知道,即为 O 所知。

在有认证服务器的情况下,使用比较广泛的密钥分配协议是 Kerberos 协议[6]。Kerberos 协议成为了事实上的计算机网络认证标准之一,该协议以 Needham-Schroeder 协议为基础,用时间戳替代了原协议中的挑战-响应部分。下面给出该协议的第 5 版本的简化形式。

协议 10.6 简化的 Kerberos 协议。

参数设置:

用户 A 和服务器 TA 的共享密钥是 K_A,用户 B 和服务器 TA 的共享密钥是 K_B,L 为使用期限,time 为时间戳。A、B 和 TA 所使用的对称密码算法的加密变换和解密变换分别记为 E 和 D。ID(A)和 ID(B)分别是 A 和 B 的标识符。

具体步骤:

(1) A 选择一个随机数 r_A,并将 ID(A)、ID(B)和 r_A 发送给 TA。

(2) TA 随机选择一个会话密钥 K 和一个有效期(或者使用期限)L。然后计算给 B 的票据 $t_B = E_{K_B}(K \| \text{ID}(A) \| L)$和 $y_1 = E_{K_A}(r_A \| \text{ID}(B) \| K \| L)$。TA 将 t_B 和 y_1 发送给 A。

(3) A 使用他的密钥 K_A 解密 y_1,得到 K。然后 A 确定当前时间 time 并计算 $y_2 = E_K(\text{ID}(A) \| \text{time})$。最后,$A$ 将 t_B 和 y_2 发送给 B。

(4) B 使用他的密钥 K_B 解密 t_B,得到 K。他还要使用密钥 K 解密 y_2 得到 time。然后 B 计算 $y_3 = E_K(\text{time}+1)$,最后,B 将 y_3 发送给 A。

与 Needham-Schroeder 协议一样,在 Kerberos 协议中需要某些有效性检验。这些检验过程如下。

(1) 当 A 解密 y_1 时,他检验明文 $D_{K_A}(y_1)$是否具有以下格式。

$D_{K_A}(y_1) = r_A \| \text{ID}(B) \| K \| L$,对于某个 K 和 L。如果这一条件不满足,则 A"拒绝"并取消当前的会话。

(2) 当 B 解密 y_2 和 t_B 时,他查看明文 $D_K(y_2)$是否具有以下格式:$D_K(y_2) = \text{ID}(A) \| \text{time}$,以及明文 $D_{K_B}(t_B)$是否为以下格式:$D_{K_B}(t_B) = K \| \text{ID}(A) \| L$,其中 ID($A$) 在两个明

文中是一样的，并且 time≤L。如果这些条件成立，那么 B“接受”；否则 B“拒绝”。

(3) 当 A 解密 y_3 时，他检查 $D_K(y_3)$=time+1。如果这一条件成立，则 A“接受”；否则 A“拒绝”。

下面对协议 10.6 的流程再做一些解释。

当 A 向 TA 发送会话密钥的请求之后，TA 随机产生一个新的会话密钥 K。同时，TA 指定一个使用期限 L，在这个期限内 K 是有效的。所有这些信息在发送给 A 之前都被加密。

A 使用他自己的密钥解密 y_1，从而得到 K 和 L。他需要检查当前时间在该密钥的使用期限之内，并且 y_1 包含了 A 的随机挑战 r_A。他也要检验 y_1 包含了 ID(B)，其中 B 是他所期望的对等通信方。这些检验防止 O 重放 TA 在以前的会话中可能传送的“旧”的 y_1。

接下来，A 传递 t_B 给 B。同时，A 使用新的会话密钥 K 加密当前时间 time 和ID(A)，然后把所生成的密文 y_2 发送给 B。

当 B 接收到 A 发来的 t_B 和 y_2 之后，他解密 t_B 得到 K、L 和 ID(A)。然后他使用新的会话密钥 K 解密 y_2，并验证从 t_B 和 y_2 中解密的 ID(A)是一样的。这使得 B 确信在 t_B 中所加密的会话密钥与用于 y_2 加密的密钥是相同的。

最后，B 使用新的会话密钥 K 加密 time+1，并把加密结果返回给 A。当 A 接收到消息 y_3 之后，他使用密钥 K 解密并验证其结果为 time+1。这使得 A 确信会话密钥 K 已经成功地传送给了 B，因为在产生消息 y_3 时需要 K。

有效期 L 的目的在于防止主动敌手存储“旧”消息并在以后某个时间重新发送，就像针对协议 10.5 的 Denning-Sacco 攻击那样。Kerberos 的缺陷之一在于网络中所有的用户需要一个同步的时钟，因为它使用了当前时间来确定一个会话密钥 K 的有效性。在实际中，要提供完美的同步性是很困难的，所以必须允许某些可变的时间。

在 ISO/IEC 11770—2 标准中，也提供了基于服务器的密钥分配协议[1]。

协议 10.7 ISO/IEC 11770—2 基于服务器的密钥分配协议。

参数设置：

用户 A 和服务器 TA 共享对称密钥 K_A，用户 B 和服务器 TA 共享对称密钥 K_B。E 是一个对称密码算法。

具体步骤：

(1) A 选择一个随机数 n_A，并将其发送给 B。

(2) B 选择一个随机数 n_B，连同从 A 得到的随机数 n_A，用户 A 的标识符，以及一个随机选择的会话密钥 k，一起用 B 与服务器 TA 共享的密钥进行加密，即计算 $E_{K_B}(n_B, n_A, A, k)$并将其发送给 TA。

(3) TA 用其与 B 共享的密钥解密数据，得到两个随机数、A 的标识符及会话密钥 k。TA 将第一个随机数连同 A 的标识符用 TA 与 B 共享的密钥加密，即计算 $E_{K_B}(n_B, A)$；将第二个随机数、会话密钥 k 以及 B 的标识符用 TA 与 A 共享的密钥加密，即计算 $E_{K_A}(n_A, k, B)$，然后将两部分数据一起发送给 B。

(4) B 收到消息，用其与 TA 共享的密钥解密第一部分消息，得到一个随机数和 A 的标识符。B 验证得到的随机数是否与自己发送的随机数一致，如果一致，他把第二部分数据 $E_{K_A}(n_A, k, B)$发送给 A。

(5) A 用其与 TA 共享的密钥解密消息,得到一个随机数、会话密钥 k 和 B 的标识符。A 验证得到的随机数是否与自己发送的随机数一致,如果一致,保存会话密钥 k。

Bellare 和 Rogaway 于 1995 年提出了一个在某些假设下可证明安全的密钥分配协议[7],下面就来描述这一协议。

协议 10.8 Bellare-Rogaway 密钥分配协议。

参数设置:

用户 A 和服务器 TA 的共享密钥是 K_A,用户 B 和服务器 TA 的共享密钥是 K_B。选定对称密码算法 E 和 MAC 算法。ID(A)和 ID(B)分别是 A 和 B 的身份标识符。

具体步骤:

(1) A 选择一个随机数 r_A,并将 ID(A)、ID(B)和 r_A 发送给 B。

(2) B 选择一个随机数 r_B,并将 ID(A)、ID(B)、r_A 和 r_B 发送给 TA。

(3) TA 随机选择一个会话密钥 K。然后计算

$$y_B = (E_{K_B}(K), \mathrm{MAC}_B(\mathrm{ID}(A) \| \mathrm{ID}(B) \| r_B \| E_{K_B}(K)))$$

和

$$y_A = (E_{K_A}(K), \mathrm{MAC}_A(\mathrm{ID}(B) \| \mathrm{ID}(A) \| r_A \| E_{K_A}(K)))$$

TA 将 y_B 发送给 B,y_A 发送给 A。

在协议 10.8 中,A 和 B 都选择随机挑战并发送给 TA。于是,在 TA 分发会话密钥之前 B 参与了协议。TA 发送给 A 的信息包括以下内容。

(1) 一个会话密钥(使用 A 的秘密密钥加密)。

(2) 一个关于加密的会话密钥、A 和 B 的身份以及 A 的挑战的 MAC。

发送给 B 的信息也是类似的。

如果各自的 MAC 是有效的,那么 A 和 B“接受”(注意到这些 MAC 是用 TA 所知道的秘密的 MAC 密钥计算出来的)。例如,当 B 接收到加密的会话密钥 $y_{B,1}$ 及 MAC$y_{B,2}$时,他验证

$$y_{B,2} = \mathrm{MAC}_B(\mathrm{ID}(A) \| \mathrm{ID}(B) \| r_B \| y_{B,1})$$

注意到,这个协议中没有提供密钥确认。例如,当 A 接受时,他并不知道 B 是否已经接受,或者甚至不知道 B 是否收到 TA 发送的消息。当 A 接受时,只是意味着他收到了预期的信息,并且该信息是有效的(或者更确切地说,MAC 是有效的)。在 A 看来,当他接受时,他相信自己收到了来自 TA 的一个新的会话密钥。此外,由于这个会话密钥使用 A 的密钥进行了加密,A 确信其他任何人都不能从他所接收到的信息中计算出会话密钥 K。当然,B 也应该收到了同一个会话密钥的一份加密。实际上,A 并不知道这件事是否已经发生,但是我们将说明 A 可以确信除了 B 之外任何人都不能计算出新的会话密钥。换句话说,我们已经从密钥分配协议中去掉了密钥确认(单方的或者双方的)这一目标。取而代之的是一个有些弱化的(但仍然有用的)目标,在这个目标中,从协议中已经“接受”的参与方的角度来看,除了他们预期的对等方,其他任何人都不能计算出新的会话密钥。

敌手的目标是导致一个诚实的参与方在某种情形下“接受”会话,但是该参与方的对等方之外的某个人知道该会话密钥的值。例如,假设诚实的 A“接受”会话,并且他的对等方是 B。称敌手 O 达到了其目的,如果他(O)可以计算出会话密钥,或者其他某个网络用户(如 C)能够计算出会话密钥。另一方面,如果 A 是唯一能够计算出会话密钥的网络用户,则不

考虑 O 的这种攻击。在这种情况下，B 不能计算出会话密钥，但是其他任何人也不能计算(除了 A)。

非正式地讲，可以定义安全的密钥分配协议为满足下面性质的密钥分配协议：如果一个参与方在一次会话中“接受”，那么该参与方的对等方之外的其他人知道会话密钥的概率很小。

下面来说明协议 10.8 是安全的。首先给出几个合理的假设，包括假设 A、B 和 TA 都是诚实的，协议中所使用的加密算法和 MAC 算法是安全的，秘密密钥只有预期的所有者才知道，并且随机挑战是使用真随机数生成器产生的。最后，假设 TA 使用真随机数生成器来产生会话密钥。

考虑 O 可以实施的不同的攻击方式。对于每一种可能的方式，可以证明：除了很小的概率之外，O 都不会成功。这些可能性并不是互相排斥的。

(1) O 是被动敌手。

(2) O 是主动敌手，A 在协议中是一个合法的参与方。O 可以冒充 B 或者 TA，并且 O 可以拦截和修改在协议中发送的消息。

(3) O 是主动敌手，B 在协议中是一个合法的参与方。O 可以冒充 A 或者 TA，并且 O 可以拦截或者修改在协议中发送的消息。

下面将针对上面的每一种攻击情形，讨论协议将产生的结果。

(1) 如果敌手是被动的，那么在 A 和 B 作为两个参与方的任何会话中，他们都会输出“接受”。更进一步，他们都能够解密出同一个会话密钥 K。其他任何人(包括 O)都不能计算 K，因为加密算法是安全的。

(2) 假设 A 在协议中是一个合法的参与方。他希望得到一个只有 B 与他才知道的新的会话密钥。但是，A 不知道他是否的确在与 B 进行通信，因为 O 可能冒充 B。当 A 收到消息 y_A 时，他检验 MAC 是否有效。这个 MAC 包括了 A 的随机挑战 r_A、A 和 B 的身份标识，以及加了密的会话密钥 $E_{K_A}(K)$。这使得 A 确信该 MAC 是由 TA 新近计算出来的，因为 TA 是除了 A 之外唯一知道 MAC_A 密钥的人。此外，随机挑战 r_A 防止了重放以前会话中的 MAC。最后，在 MAC 中包括 $E_{K_A}(K)$防止了敌手把 TA 选择的会话密钥替换为其他的东西。因此，A 可以确信 B(他预期的对等方)是唯一的可以解密出会话密钥 K 的其他用户，即使 O 在当前的会话中冒充 B。

(3) 假设 B 在协议中是一个合法的参与方。他相信可以得到一个只有 A 和他自己才知道的新的会话密钥。但是，B 不知道他是否的确在与 A 进行通信，因为 O 可能冒充 A。当 B 接收到消息 y_B 时，他查看 MAC 是否有效。这个 MAC 包括了 B 的随机挑战 r_B、A 和 B 的身份标识，以及加了密的会话密钥 $E_{K_B}(K)$。这使得 B 确信该 MAC 是由 TA 新近计算出来的，因为 TA 是除了 B 之外唯一知道 MAC_B 密钥的人。此外，随机挑战 r_B 防止了重放以前会话中的 MAC。最后，在 MAC 中包括 $E_{K_B}(K)$ 防止了敌手把 TA 选择的会话密钥替换为其他的东西。因此，B 可以确信 A(他预期的对等方)是唯一的可以解密出会话密钥 K 的其他用户，即使 O 在当前的会话中冒充 A。

10.1.2 基于公钥密码算法的密钥分配协议

使用公钥密码算法的密钥分配协议一般是基于公钥加密来实现的，一个参与方选择一

个会话密钥,然后使用另一方的公钥将其加密,并发送给另一方。基于公钥密码算法的协议一般都能提供认证功能。

基于公钥密码算法的密钥分配协议中,最简单的协议就是单步密钥分配协议。

协议 10.9 单步密钥分配协议。

参数设置:

用户 A 预先拥有用户 B 的加密公钥。

具体步骤:

A 选定会话密钥 k,然后用 B 的公钥加密,即计算 $E_B(k)$并发送给 B。

使用协议 10.9,A 不能获得 B 的实体认证,但是可以获得密钥认证,即除了 B 之外,任何人不能恢复会话密钥 k。

只使用公钥加密算法的密钥分配协议,也可以提供相互的实体认证和相互的密钥认证,一个例子就是 Needham-Schroeder 公钥协议[4]。

协议 10.10 Needham-Schroeder 公钥协议。

参数设置:

用户 A 拥有用户 B 的加密公钥,用户 B 拥有用户 A 的加密公钥。

具体步骤:

(1) A 选取一个随机数 n_A,用 B 的公钥加密,即计算 $E_B(n_A, A)$并将其发送给 B。

(2) B 用自己的私钥解密消息,得到随机数 n_A 和 A 的标识符。B 选取一个随机数 n_B,连同 n_A 一起用 A 的公钥加密,即计算 $E_A(n_A, n_B)$并发送给 A。

(3) A 用自己的私钥解密消息,得到两个随机数,并检查第一个随机数是否与自己发送的随机数一致。如果一致,A 将第二个随机数用 B 的公钥加密,即计算$E_B(n_B)$并将其发送给 B。

(4) B 用自己的私钥解密消息,得到一个随机数,并检查该随机数是否与自己发送的随机数一致。

使用协议 10.10,A 和 B 的会话密钥可以通过计算一个以 n_A 和 n_B 为变量的单向函数来实现。这个协议提出之后,人们对这个协议一直很感兴趣。在 1996 年,Lowe 提出了对这个协议的一个攻击方法[8]:在 A 发起与 C 的 Needham-Schroeder 公钥协议时,C 可以冒充 A 与 B 完成 Needham-Schroeder 公钥协议。

在密钥分配协议中,除了要保证密钥信息的机密性外,有时还需要信息源认证,此时一般要同时使用加密技术和签名技术,加密技术提供机密性,签名技术提供信息源认证。在现有的使用公钥密码技术的密钥分配协议中,大部分协议都同时使用加密技术和签名技术。

协议 10.11 ISO/IEC 11770—3 密钥分配协议[9]。

参数设置:

用户 A 拥有用户 B 的加密公钥,用户 B 拥有用户 A 的加密公钥。

具体步骤:

(1) A 选择一个随机数 n_A,并将其发送给 B。

(2) B 收到消息后,首先选择一个随机数 n_B 以及一个会话密钥 k。然后 B 用 A 的公钥将自己的标识符和自己选取的会话密钥 k 加密,即计算 $E_A(B, k)$;随后将 A 的标识符以及两个随机数连同加密后的信息一起用自己的私钥签名,即计算 $\mathrm{Sig}_B(A, n_A, n_B, E_A(B,k))$;

最后，B 将签名的信息以及签名的结果一起发送给 A。

(3) A 收到消息后，用 B 的公钥验证签名，并检查第一个随机数是否与自己发送的随机数一致。如果通过验证，并且随机数与自己发送的一致，A 接受会话密钥 k。

在这个协议中，A 使用随机数 n_A 来保证密钥的新鲜性及对 B 的实体认证。只要 B 是可信的，A 就可以认为得到的密钥是安全的，并完成密钥认证。对于 B，可以保证密钥是安全的，但是不能对用户进行实体认证。在该协议中，随机数 n_B 是可选的。如果删除这一项，Shoup 证明该协议在他的模拟模型下仍然是安全的[10]。

在另一个比较常用的 X.509 标准[11]中，也同时使用加密技术和签名技术来实现密钥分配协议。

协议 10.12 X.509 三向认证协议。

参数设置：

每个用户都拥有自己的用于签名和加密的公、私钥对，并且每个用户预先获得另一个用户的加密公钥。

具体步骤：

(1) A 首先选取一个随机数 n_A、一个会话密钥 k_{AB}，并获得时间戳 T_A。然后，A 将会话密钥 k_{AB} 用 B 的公钥加密，随后将时间戳 T_A、随机数 n_A、B 的标识符以及加密的信息一起用自己的私钥签名，即计算 $\text{Sig}_A(T_A, n_A, B, E_B(k_{AB}))$。最后，$A$ 将签名的信息及签名的结果一起发送给 B。

(2) B 收到消息后，首先用 A 的公钥验证签名。如果通过验证，B 选取一个随机数 n_B、一个会话密钥 k_{BA}，并获得时间戳 T_B。然后，B 将会话密钥 k_{BA} 用 A 的公钥加密，随后将时间戳 T_B、随机数 n_B、A 的标识符、A 选择的随机数 n_A 及加密的信息一起用自己的私钥签名，即计算 $\text{Sig}_B(T_B, n_B, A, n_A, E_A(k_{BA}))$。最后，$B$ 将签名的信息及签名的结果一起发送给 A。

(3) A 收到消息后，首先用 B 的公钥验证签名，并检查第二个随机数是否与自己发送的随机数一致。如果通过验证，并且随机数与自己发送的一致，A 将接收到的消息中的第一个随机数以及 B 的标识符一起用自己的私钥签名，即计算 $\text{Sig}_A(n_B, B)$。最后，A 将签名的信息及签名的结果一起发送给 B。

(4) B 收到消息后，首先用 A 的公钥验证签名，并检查随机数是否与自己发送的随机数一致。

在这个协议中，最终的会话密钥可以通过计算一个以 k_{AB} 和 k_{BA} 为变量的单向函数得到。由于最终会话密钥由两个参与方共同决定，这个协议应该认为是密钥协商协议，而不是密钥分配协议。协议最终得到的会话密钥对于 A 和 B 来说，都保证是新鲜的。如果协议只执行第一步，得到的是 X.509 单向认证协议。在 X.509 单向认证协议中，会话密钥由一个参与方选定，然后发送给另一个参与方，是一个密钥分配协议。这个密钥分配协议，会话密钥只是对参与方 A 来说是新鲜的。

除了上面两个标准外，同时使用加密技术和签名技术的密钥分配协议还有很多，再举一个例子：Blake-Wilson 和 Menezes 可证明安全密钥分配协议[12]。

协议 10.13 Blake-Wilson 和 Menezes 可证明安全密钥分配协议。

参数设置：

用户 A 拥有用户 B 的加密公钥，用户 B 拥有用户 A 的加密公钥。

具体步骤：

(1) A 选择一个随机数 n_A，连同自己的标识符一起发送给 B。

(2) B 收到消息后，首先选择一个随机数 n_B、一个会话密钥 k。然后，B 将自己的标识符和会话密钥 k 一起用 A 的公钥加密，随后将 B 的标识符、A 的标识符、自己选择的随机数 n_B、接收到的随机数 n_B 及加密的信息一起用自己的私钥签名，即计算 $\mathrm{Sig}_B(B, A, n_B, n_A, E_A(B, k))$。最后，$B$ 将签名的信息及签名的结果一起发送给 A。

(3) A 收到消息后，首先用 B 的公钥验证签名，并检查第二个随机数是否与自己发送的随机数一致。如果通过验证，并且随机数与自己发送的一致，A 将 A 的标识符、B 的标识符和接收到消息中的第一个随机数一起用自己的私钥签名，即计算 $\mathrm{Sig}_A(A, B, n_B)$。最后，$A$ 将签名的信息及签名的结果一起发送给 B。

(4) B 收到消息后，用 A 的公钥验证签名，并检查随机数是否与自己发送的随机数一致。

这个协议在 Bellare-Rogaway 模型下可证明是安全的。

使用公钥密码技术的密钥分配协议中，还有一些协议，除了使用公钥加密技术和数字签名技术外，还使用了对称加密技术，如 Beller-Yacobi 协议[13]。

协议 10.14 Beller-Yacobi 协议。

参数设置：

每个用户都拥有自己的用于签名和加密的公、私钥对。E 是对称密码算法。

具体步骤：

(1) A 将自己的标识符连同自己的公钥一起发送给 B。

(2) B 收到消息后，选取一个随机的会话密钥 k，将这个会话密钥 k 用 A 的公钥加密，即计算 $E_A(k)$ 并发送给 A。

(3) A 收到消息后，首先用自己的私钥解密消息，得到会话密钥 k，然后 A 选取一个随机数 n_A，用会话密钥 k 将这个随机数加密，即计算 $E_k(n_A)$ 并发送给 B。

(4) B 收到消息后，首先用会话密钥 k 解密消息得到随机数 n_A，并用自己的私钥对随机数 n_A 签名。然后，B 将自己的标识符、自己的公钥、自己的证书及签名的结果一起用会话密钥加密，即计算 $E_k(B, K_B, \mathrm{Cert}(B), \mathrm{Sig}_B(n_A))$ 并发送给 A。

(5) A 收到消息后，首先用会话密钥 k 解密消息，得到 B 的标识符、B 的公钥、B 的证书以及 B 对自己发送的随机数的签名。A 用 B 的公钥验证签名信息的正确性。

这个协议的独特之处是协议双方可以使用不同的公钥系统，如果一方的计算能力有限，可以使用需要计算资源较小的算法。后来，Boyd 和 Mathuria 发现了这个协议的一个潜在攻击方法[14]，一个攻击者可以冒充某个用户与另一个用户建立会话密钥。

10.2 密钥协商协议

密钥协商协议大部分使用公钥密码算法来实现，也有少部分使用对称密码算法来实现，一般需要一个密钥预分配协议。密钥预分配协议是提供共享密钥的一种方法，该方法在初始阶段由一个可信服务器为用户生成并分发秘密信息，以使任何一对用户可以随后计算其他所有人(服务器除外)不知道的共享密钥。使用密钥预分配方法的协议严格上讲并不能称

为密钥协商协议，但是，这类协议与静态的 Diffie-Hellman 协议类似，一般情况下就把这类协议也视为密钥协商协议。

基于公钥密码算法的密钥协商协议的研究成果相当丰富。本节只介绍一些比较基本的密钥协商协议，这些协议大部分是以 Diffie-Hellman 密钥协商协议为基础的。而对于基于身份的密钥协商协议、基于口令的密钥协商协议、群组密钥协商协议和跨域密钥协商协议，将在随后的几节中介绍。

10.2.1 Diffie-Hellman 密钥预分配协议

Diffie-Hellman 密钥预分配协议[15]是 10.2.4 小节将要介绍的 Diffie-Hellman 密钥协商协议的一种修改。该协议在假定与离散对数问题相关的一个问题是难处理的情况下是计算安全的。

下面仅描述在 Z_p 上的一个协议，p 是一个素数。不过，该协议可在计算离散对数问题时难处理的任何有限群上实现。

协议 10.15 Diffie-Hellman 密钥预分配协议。

参数设置：

假定 α 是 Z_p 的一个生成元（也称本原元），网络中的任何用户都知道 p 和 α 的值。用 ID(U)表示网络中的用户 U 的某些识别信息，诸如姓名、E-mail 地址、电话号码或别的有关信息。每个用户 U 有一个秘密指数 a_U（$0 \leqslant a_U \leqslant p-2$）和一个相应的公钥 $b_U = \alpha^{a_U} \bmod p$。

可信中心 TA 有一个签名算法，该签名算法的公开验证算法记为 Ver_{TA}，秘密签名算法记为 Sig_{TA}。根据该协议，在签名消息之前，总先将消息用一个公开的 Hash 函数杂凑。但为了简单起见，这里将略去这一步。

当一个用户 U 入网时，可信中心 TA 需给他颁发一个证书（Certificate）。用户 U 的证书为：$C(U) = (\mathrm{ID}(U), b_U, \mathrm{Sig}_{TA}, (\mathrm{ID}(U), b_U))$。可信中心无需知道 a_U 的值。证书可存储在一个公开的数据库中，也可由用户自己存储。网络中的任何人能验证可信中心对证书的签名。Diffie-Hellman 密钥预分配协议的目的是使通信双方建立共同的密钥。

具体步骤：

(1) U 将他的证书 $C(U)$ 发送给 V。

(2) V 验证 U 的证书并使用他自己的秘密值 a_V 及从 U 的证书中获得的公开值 b_U 计算 $b_U^{a_V} = \alpha^{a_U b_V} \bmod p$，从而得到 $k_{U,V}$。

(3) V 将他的证书 $C(V)$ 发送给 U。

(4) U 验证 V 的证书并使用他自己的秘密值 a_U 及从 V 的证书中获得的公开值 b_V，计算 $b_V^{a_U} = \alpha^{a_U b_V} \bmod p$，从而得到 $k_{U,V}$。

关于 Diffie-Hellman 的密钥预分配协议的安全性，要考虑对它的几种攻击。如果敌手发起的是主动攻击，企图篡改 U 的证书中的消息 b_U，那么他难以通过对 TA 签名的验证。如果敌手发起的被动攻击，企图计算 $k_{U,V}$，一种办法是求出 a_U，从而得到 $k_{U,V} = b_V^{a_U} \bmod p$，但求 a_U 要计算离散对数问题，按假设这是难问题。实际上，$k_{U,V} = b_U^{\log_\alpha b_V} \bmod p$，已知 b_U、b_V、p、α 求 $k_{U,V}$ 被称为 Diffie-Hellman 问题，敌手被动攻击能否成功等价于 Diffie-Hellman 问题能否求解。人们猜测解 Diffie-Hellman 问题的任何算法也可用来解离散对数问题。但至今这个猜测还未得到证明。虽然不能精确地说 Diffie-Hellman 问题有多难，但可以将这

个问题和第 2 章介绍的 ElGamal 加密算法的安全性联系起来。

定理 10.1 破译 ElGamal 加密算法等价于解 Diffie-Hellman 问题。

证明 设 ElGamal 加密算法的秘密密钥为 a，公钥为 $\beta=\alpha^a \bmod p$、p 和 α，其中 p 和 α 与 Diffie-Hellman 问题中的 p 和 α 相同。

对消息 $x\in Z_p$ 的加密过程如下。

随机选择一个数 $k\in Z_{p-1}$，$E_k(x,k)=(y_1,y_2)$，这里 $y_1=\alpha^k \bmod p$，$y_2=x\beta^k \bmod p$。

解密过程为：$x=D_k(y_1,y_2)=y_2(y_1^a)^{-1} \bmod p$，$y_1,y_2\in Z_p^*$。

假定有一个算法 A，A 能解 Diffie-Hellman 问题，并给定一个用 ElGamal 加密算法加密的密文 (y_1,y_2)。将 p、α、y_1 和 β 作为算法 A 的输入，则获得 $A(p,\alpha,y_1,\beta)=A(p,\alpha,\alpha^k,\alpha^a)=\alpha^{ka} \bmod p=\beta^k \bmod p$。此时密文 (y_1,y_2) 对应的明文可由下式容易求出：$x=y_2(\beta^k)^{-1} \bmod p$。

反之，假定有一个算法 B，B 能完成 ElGamal 加密算法的解密过程。也就是说，如果将 p、α、β、y_1 和 y_2 作为 B 的输入，那么能求得 $x=B(p,\alpha,\beta,y_1,y_2)=y_2(y_1^{\log_\alpha \beta})^{-1} \bmod p$。现在对任何 $\gamma\in Z_p^*$，利用算法 B 可以计算出 $\gamma^{\log_\alpha \beta}$：$B(p,\alpha,\beta,\gamma,1)^{-1}=(1(\gamma^{\log_\alpha \beta})^{-1})^{-1}=\gamma^{\log_\alpha \beta} \bmod p$。

10.2.2 Blom 密钥预分配协议

假定有一个拥有 n 个用户的网络，首先给出一个“平凡的”无条件安全的密钥预分配协议。对每对用户 $\{U,V\}$，TA 选择一个随机的密钥 $K_{U,V}=K_{V,U}$ 并把它通过“离线”的安全信道传送给 U 和 V，即密钥的传输不在网络上进行，因为网络是不安全的。遗憾的是，该协议中的每个用户必须存储 $n-1$ 个密钥，而且 TA 需要安全地传送总共 $\binom{n}{2}$ 个密钥，这就是人们通常所说的 n^2 问题。即使较小的网络，这个代价都高得惊人，所以这不是一个实用的解决办法。因此，尽可能减少需要传输和存储的信息数量，并且仍然使得每对用户 U 和 V 能够独立地计算一个秘密的密钥 $K_{U,V}$ 是很有意义的。下面将要介绍的 Blom 密钥预分配协议[16]就是一个可以满足这些要求的优美协议。

这里简单地讨论一下在无条件安全的密钥预分配协议研究中使用的安全模型。假定 TA 向 n 个网络用户安全地分发秘密信息。敌手可以收买至多包括 k 个用户的用户子集，并且得到其所有的秘密信息，其中 k 是预先指定的安全参数。敌手的目标是确定一对未被收买的用户的秘密的长期密钥，即 LL 密钥。Blom 密钥预分配协议是无条件安全地抵抗这类敌手的密钥预分配协议。

每对用户 U 和 V 期望能够计算出一个密钥 $K_{U,V}=K_{V,U}$。所以，这里的安全要求是：任何与 $\{U,V\}$ 不相交的至多包括 k 个用户的集合都不能确定有关 $K_{U,V}$ 的任何信息（这里所说的是指无条件安全性）。

在 Blom 密钥预分配协议中，密钥取自于有限域 $\mathbb{Z}_p$，其中 $p\geqslant n$ 是素数。TA 通过一个安全信道向每个用户发送 $k+1$ 个 $\mathbb{Z}_p$ 中的元素（与本节开头介绍的“平凡的”密钥预分配协议中的 $n-1$ 个元素相对），TA 所传送的信息数量与 n 是无关的。

首先描述 Blom 密钥预分配协议当 $k=1$ 时的特殊情况。这时，TA 通过一个安全信道向每个用户发送两个 $\mathbb{Z}_p$ 中的元素。要达到的安全目标是任何单个的用户，比如说 W，不能

确定有关 $K_{U,V}$ 的任何信息，只要 $W\neq U,V$。

协议 10.16 Blom 密钥预分配协议（$k=1$）。

参数设置：

TA 公开一个素数 p，每个用户 U 公布一个元素 $r_U\in\mathbb{Z}_p$，不同用户的 r_U 互不相同。

具体步骤：

(1) TA 秘密地选择 3 个随机元素 $a,b,c\in\mathbb{Z}_p$（未必不同），并构造下述对称多项式：

$$f(x,y)=a+b(x+y)+cxy \bmod p$$

(2) TA 为每个用户 U 计算多项式 $g_U(x)=f(x,r_U) \bmod p=a_U+b_Ux$，并通过安全信道把 $g_U(x)$ 发送给 U，这里 $a_U=a+br_U \bmod p$，$b_U=b+cr_U \bmod p$。

(3) 如果 U 和 V 想通信，U 计算 $K_{U,V}=g_U(r_V)$，V 计算 $K_{V,U}=g_V(r_U)$，于是他们拥有了共同的密钥

$$K_{U,V}=K_{V,U}=f(r_U,r_V)=a+b(r_U+r_V)+cr_Ur_V \bmod p$$

协议 10.16 的一个重要性质是多项式 f 是对称的，即对于所有的 x、y，$f(x,y)=f(y,x)$。这一性质保证了 $g_U(r_V)=g_V(r_U)$，所以 U 和 V 可以在协议的第(3)步计算出相同的密钥。

现在证明：没有一个用户能确定另两个用户所共有的密钥的任何信息。

定理 10.2 $k=1$ 时的 Blom 密钥预分配协议对任何单个用户是无条件安全的，但是面对两个用户合伙是不安全的。

证明 设 U、V 是任意给定的两个不同用户，第三个用户 W 想计算密钥 $K_{U,V}=a+b(r_U+r_V)+cr_Ur_V \bmod p$。$W$ 所掌握的关于 $K_{U,V}$ 的全部信息可表示为下列矩阵方程：

$$\begin{bmatrix}1 & r_U+r_V & r_Ur_V\\ 1 & r_W & 0\\ 0 & 1 & r_W\end{bmatrix}\begin{bmatrix}a\\ b\\ c\end{bmatrix}=\begin{bmatrix}K_{U,V}\\ a_W\\ b_W\end{bmatrix}$$

其中 a、b、c 和 $K_{U,V}$ 是未知数。

系数矩阵的行列式为：$r_W^2+r_Ur_V-(r_U+r_V)r_W=(r_W-r_U)(r_W-r_V)\neq 0$，任意设置 $K_{U,V}$ 的一个值 $l\in Z_p$ 上述矩阵方程关于 a、b、c 都有唯一的一个解。换句话说，W 无法判断 Z_p 中的哪一个更像 $K_{U,V}$。

但若 W 与另一用户 X 合伙，W 与 X 根据共同掌握的信息，可以写出下列矩阵方程：

$$\begin{bmatrix}1 & r_X & 0\\ 1 & r_W & 0\\ 0 & 1 & r_W\end{bmatrix}\begin{bmatrix}a\\ b\\ c\end{bmatrix}=\begin{bmatrix}a_X\\ a_W\\ b_W\end{bmatrix}$$

注意系数矩阵行列式为 $(r_W-r_X)r_W\neq 0$，由该方程可求解 (a,b,c)，因此，能够构建多项式 $f(x,y)$ 并可计算出任何想要的密钥，当然可以计算出 $K_{U,V}=a+b(r_U+r_V)+cr_Ur_V$。

规模为 1 的 Blom 密钥预分配协议可直接推广为规模为 $k(1\leqslant k\leqslant n-2)$ 的 Blom 密钥预分配协议。唯一需要改变的是在第(1)步 TA 改用以下形式的秘密的一个对称多项式：

$$f(x,y)=\sum_{i=0}^{k}\sum_{j=0}^{k}a_{i,j}x^iy^j \bmod p$$

其中 $a_{i,j}\in\mathbb{Z}_p(0\leqslant i\leqslant k,0\leqslant j\leqslant k)$，并且对于所有的 i,j 有 $a_{i,j}=a_{j,i}$。协议的其余步骤不需要改变。

协议 10.17 Blom 密钥预分配协议(任意的 k)。

参数设置：

TA 公开一个素数 p,每个用户 U 公布一个元素 $r_U \in \mathbb{Z}_p$,不同用户的 r_U 互不相同。

具体步骤：

(1) 对于 $0 \leqslant i, j \leqslant k$,TA 选择随机元素 $a_{i,j} \in \mathbb{Z}_p$ 使得对所有的 i, j, $a_{i,j} = a_{j,i}$。TA 构造对称多项式

$$f(x,y) = \sum_{i=0}^{k} \sum_{j=0}^{k} a_{i,j} x^i y^j \bmod p$$

(2) TA 为每个用户 U 计算多项式

$$g_U(x) = f(x, r_U) \bmod p = \sum_{i=0}^{k} a_{U,i} x^i$$

并通过安全信道把系数向量$(a_{U,0}, \cdots, a_{U,k})$发送给 U。

(3) 对于任何两个用户 U 和 V,其密钥为 $K_{U,V} = f(r_U, r_V)$,其中 U 计算 $K_{U,V} = g_U(r_V)$,而 V 计算 $K_{V,U} = g_V(r_U)$。

下面将说明 Blom 密钥预分配协议满足以下安全属性。

(1) 不存在 k 个用户的集合,如$\{W_1, \cdots, W_k\}$,可以确定出其他两个用户的密钥的任何信息,如 $K_{U,V}$。

(2) 任何 $k+1$ 个用户的集合,如$\{W_1, \cdots, W_{k+1}\}$,可以攻破该协议。

可以通过修改当 $k=1$ 时给出的方法证明协议 10.17 满足上述两个安全属性。这里介绍另一种方法,该方法中使用了拉格朗日插值公式和二元拉格朗日插值公式。拉格朗日插值公式已在 7.2 节中介绍,这里介绍一下二元拉格朗日插值公式。设 p 是素数,$x_1, x_2, \cdots, x_{m+1}$ 是$\mathbb{Z}_p$中不同的元素,设 $a_1(x), a_2(x), \cdots, a_{m+1}(x) \in \mathbb{Z}_p[x]$是次数至多为 m 的多项式。存在次数至多为 m 的唯一的多项式$A(x,y) \in \mathbb{Z}_p[x,y]$(以 x 和 y 为变元),使得 $A(x, y_i) = a_i(x)$($1 \leqslant i \leqslant m+1$)。并且多项式 $A(x,y)$为

$$A(x,y) = \sum_{j=1}^{m+1} a_j(x) \prod_{1 \leqslant h \leqslant m+1, h \neq j} \frac{y - y_h}{y_j - y_h}$$

上述公式比较容易证明,请读者作为练习自行完成。

现在来证明下面的定理。

定理 10.3 Blom 密钥预分配协议是无条件安全地抵抗 k 个用户的攻击,但是任何 $k+1$ 个用户都能够攻破协议。

证明 易知,如果存在 $k+1$ 个恶意的用户,那么 Blom 密钥预分配协议是不安全的。$k+1$个恶意用户的集合,如 $W_1, \cdots, W_{k+1}$,联合起来知道 $k+1$ 个次数为 k 的多项式,即

$$g_{W_i}(x) = f(x, r_{W_i}) \bmod p$$

其中 $1 \leqslant i \leqslant k+1$。利用二元插值公式,他们可以计算出 $f(x,y)$,因此,他们就能够计算出想要的任何密钥 $K_{U,V}$。

接下来证明 Blom 密钥预分配协议可以无条件安全地抵抗 k 个恶意用户的联合攻击。一个具有 k 个恶意用户的集合,如 $W_1, \cdots, W_k$,联合起来掌握 k 个次数为 k 的多项式,即

$$g_{W_i}(x) = f(x, r_{W_i}) \bmod p$$

其中 $1 \leqslant i \leqslant k$。下面来说明这些信息与密钥的任何可能取值都是一致的。设 K 是实际的

密钥(其值不为合谋者所知),而 K^* 是任意的值。下面证明存在对称多项式 $f^*(x,y)$,它与合谋者所知道的信息是一致的,并且使得与 $f^*(x,y)$ 相关的密钥是 K^*。从而,联合攻击并不能排除任何可能的密钥值。

按照以下方式定义多项式 $f^*(x,y)$:

$$f^*(x,y) = f(x,y) + (K^* - K)\prod_{1\leqslant i\leqslant k}\frac{(x-r_{W_i})(y-r_{W_i})}{(r_U-r_{W_i})(r_V-r_{W_i})} \tag{10-1}$$

$f^*(x,y)$ 具有以下一些性质。

(1) f^* 是一个对称多项式,即 $f^*(x,y)=f^*(y,x)$,这是因为 $f(x,y)$ 是对称的,而且式(10-1)中的乘积关于 x 和 y 也是对称的。

(2) 对于 $1\leqslant i\leqslant k$,有

$$f^*(x,r_{W_i}) = f(x,r_{W_i}) = g_{W_i}(x)$$

这是因为当 $y=r_{W_i}$ 时式(10-1)的乘积中包括一个等于 0 的项,从而乘积为 0。

(3) $f^*(r_U,r_V)=f(r_U,r_V)+K^*-K=K^*$,因为此时式(10-1)中的乘积等于 1。

这 3 个性质表明,对于密钥的任何可能的取值 K^*,都存在一个对称多项式 $f^*(x,y)$ 使得 $f^*(U,V)=K^*$,而 k 个恶意用户所掌握的秘密信息保持不变。

Blom 密钥预分配协议的一个缺点是存在一个必须事先指定的苛刻的安全门限(即 k 值)。一旦多于 k 个用户决定联合,整个协议将被攻破。但 Blom 密钥预分配协议在存储需求方面来说是最优的。已经证明,在任何可以抵抗 k 个用户联合的无条件安全的密钥预分配协议中,每个用户的存储至少是密钥长度的 $k+1$ 倍。

10.2.3 Leighton Micali 密钥协商协议

Leighton 和 Micali 提出了一个相对简单的密钥协商协议[17]。在该协议中,用户的密钥存储在一个抗篡改的设备中,这个假定可以减小合谋攻击的威胁。

协议 10.18 Leighton-Micali 密钥协商协议。

参数设置:

系统包括若干个用户和一个可信服务器 TA。每个用户生成一个身份比特串(a_1, a_2, …, a_n),这个比特串由一个以用户的身份标识符为变量的公开函数计算,h 为一个单向函数。

具体步骤:

(1) TA 随机生成 n 个秘密值(X_1, X_2, …, X_n)。

(2) TA 为身份比特串为(a_1, a_2, …, a_n)的用户分配一个秘密序列(t_1, t_2, …, t_n),这里,如果 $a_i=0$,则 $t_i=X_i$;如果 $a_i=1$,则 $t_i=h(X_i)$。

(3) 两个身份比特串分别为(a_1, a_2, …, a_n)和(b_1, b_2, …, b_n)的用户,通过下面方式计算它们的共享秘密 $k=(k_1, k_2, \cdots, k_n)$:如果 $a_i=b_i=0$,则 $k_i=X_i$;否则,$k_i=h(X_i)$。

10.2.4 Diffie-Hellman 密钥协商协议

第一个也是最著名的密钥协商协议是 Diffie-Hellman 密钥协商协议[18]。

协议 10.19 与协议 10.15 是非常类似的,差别仅在于每一次协议运行时,用户 U 和 V 都会分别选取一个新的指数 a_U 和 a_V,并且在该协议中没有长期密钥。

协议 10.19 Diffie-Hellman 密钥协商协议。

参数设置：

公开群$(G,\cdot)$和阶为 n 的元素 $\alpha\in G$。

具体步骤：

(1) U 选取一个随机数 a_U，$0\leqslant a_U\leqslant n-1$，然后计算 $b_U=\alpha^{a_U}$，并将 b_U发送给 V。

(2) V 选取一个随机数 a_V，$0\leqslant a_V\leqslant n-1$，然后计算 $b_V=\alpha^{a_V}$，并将 b_V发送给 U。

(3) U 计算得出 $K=(b_V)^{a_U}$，V 计算得出 $K=(b_U)^{a_V}$。

在 Diffie-Hellman 密钥协商协议的一个会话结束时，用户 U 和 V 计算出了同一个密钥：

$$K=\alpha^{a_U a_V}=\mathrm{CDH}\,(\alpha,b_U,b_V)$$

这里，CDH 通常指的是计算 Diffie-Hellman 问题。因为假定判定性 Diffie-Hellman 问题是困难的，所以一个被动的敌手无法计算出关于 K 的任何信息。

由协议 10.19 的流程可知，Diffie-Hellman 密钥协商协议的基本模式如图 10.1 所示。

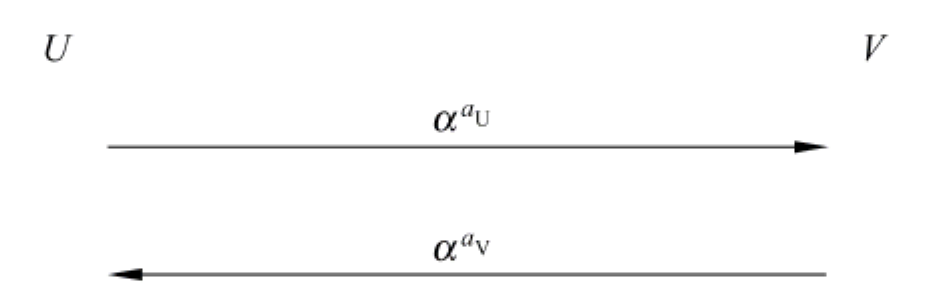


图 10.1 Diffie-Hellman 密钥协商协议基本模式

不幸的是，这种协议很容易遭受主动敌手发起的中间入侵攻击。其基本模式如图 10.2 所示，对 Diffie-Hellman 密钥协商协议的中间入侵攻击来讲，W 将截取 U 和 V 之间的信息并替换成自己的信息。在会话结束时，U 与 W 确立了秘密密钥 $\alpha^{a_U a'_V}$，V 与 W 确立了秘密密钥 $\alpha^{a'_U a_V}$。当 U 要加密一条信息后发送给 V 时，W 可以对它进行解密而 V 却不可以。对于 V 发送信息给 U 的情况也是类似的。

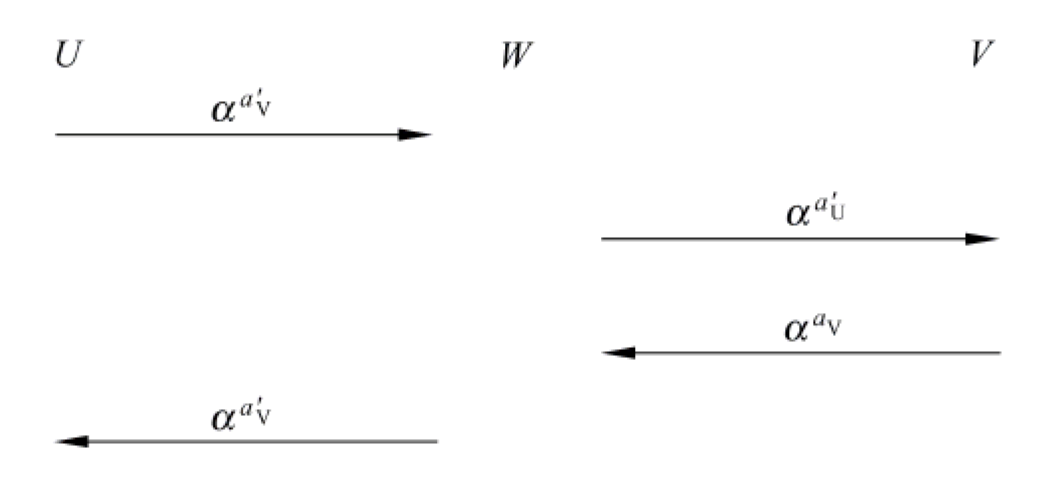


图 10.2 中间入侵攻击

显然，U 和 V 有必要保证是在向对方而不是 W 来交换信息(包括密钥)。在交换密钥之前，U 和 V 要执行一个单独的协议来确定彼此的身份，如可以使用一个安全的交互认证协议。但这仍不能抵抗中间入侵攻击。这是因为 W 可以等到 U 和 V 相互证明身份之后进行中间入侵攻击。有一个更有效的方法可以用来设计密钥协商协议，即在密钥确定的同时就要认证参与者的身份。这种类型的密钥协商协议被称为认证密钥协商协议。

一个认证密钥协商协议可以非正式地定义为满足以下性质的密钥协商协议。

(1) 交互识别。是一个安全的交互识别协议，即在主动敌手的任何流程后，没有一个诚

实的参与者会“接受”。

(2) 密钥协商。如果不存在主动敌手，则双方参与者计算出相同的新会话密钥 K。除此之外，一个被动敌手将计算不出关于 K 的任何信息。

10.2.5 端-端密钥协商协议

本小节将描述一个认证密钥协商协议[19]，它是 Diffie-Hellman 密钥协商协议的一个改进。该协议使用了通常由可信中心 TA 签过名的证书。该认证密钥协商协议也被称作端-端密钥协商协议(简写为 STS)，是由 Diffie、Van Oorschot 和 Wiener 提出的。协议 10.20 是一个略微简化的端-端协议，它遵从 ISO 9798—3 标准的要求。其基本思想是将 Diffie-Hellman 密钥协商协议和一个安全的身份识别协议结合在一起，这里指数值 b_U 和 b_V 充当身份识别协议的随机因子。大致来讲，对随机因子进行签名提供了交互式认证。更进一步，根据 Diffie-Hellman 密钥协商协议计算出的随机因子使得用户 U 和 V 可以计算出相同的密钥 $K=\mathrm{CDH}(\alpha,b_U,b_V)$。

协议 10.20 简化的端-端密钥协商协议。

参数设置：

公开群 $(G,\cdot)$ 和一个阶为 n 的元素 $\alpha\in G$。用户 U 的签名算法记为 Sig_U，验证算法记为 Ver_U。可信中心 TA 的公开验证算法记为 Ver_{TA}。每个用户都有一个证书 $\mathrm{Cert}(U)=(\mathrm{ID}(U),\mathrm{Ver}_U,\mathrm{Sig}_{TA}(\mathrm{ID}(U),\mathrm{Ver}_U))$，这里 $\mathrm{ID}(U)$ 是 U 的识别信息。用户 V 拥有与用户 U 同样的参数。

具体步骤：

(1) U 选择一个随机数 a_U，$0\leqslant a_U\leqslant n-1$，然后计算 $b_U=\alpha^{a_U}$，并将 $\mathrm{Cert}(U)$ 和 b_U 发送给 V。

(2) V 选择一个随机数 a_V，$0\leqslant a_V\leqslant n-1$，然后计算 $b_V=\alpha^{a_V}$，$K=(b_U)^{a_V}$ 和 $y_V=\mathrm{Sig}_V(\mathrm{ID}(U)\parallel b_V\parallel b_U)$，并将 $\mathrm{Cert}(V)$、b_V 和 y_V 发送给 U。

(3) U 使用 Ver_V 来验证 y_V。如果签名 y_V 无效，则他会“拒绝”并退出；否则，他会“接受”，计算 $K=(b_V)^{a_U}$ 和 $y_U=\mathrm{Sig}_U(\mathrm{ID}(V)\parallel b_U\parallel b_V)$，并将 y_U 发送给 V。

(4) V 使用 Ver_U 验证 y_U。如果签名 y_U 无效，则他会“拒绝”；否则，他会“接受”。

现在来讨论简化的 STS 协议的安全性。为了讨论方便，将协议的一次会话中交换的信息(包括证书)绘制成图，如图 10.3 所示。

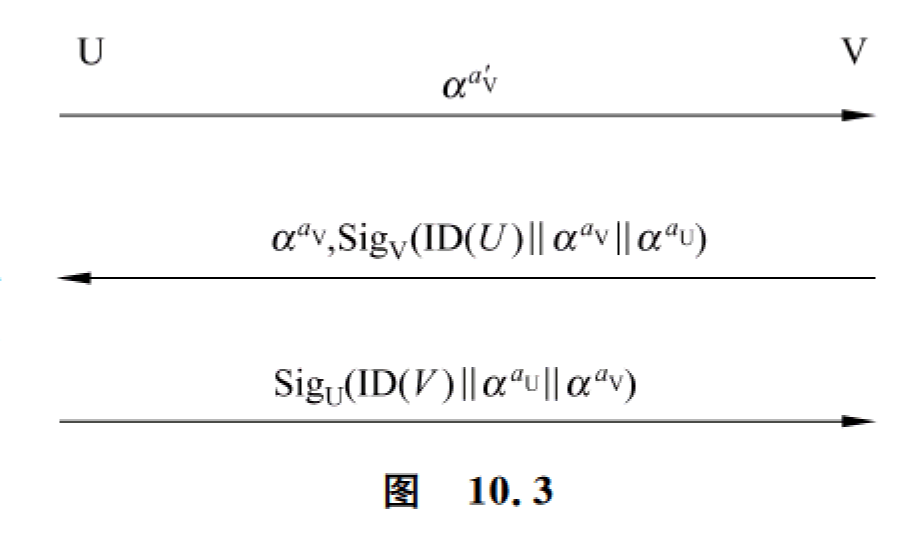


图 10.3

首先看一下如何利用签名来阻止上面所说的中间入侵攻击。假设 W 截取了 α^{a_U} 并将它替换为 $\alpha^{a'_U}$。然后 W 收到从 V 那里发送来的 α^{a_V} 和 $\mathrm{Sig}_V(\mathrm{ID}(U)\parallel\alpha^{a_V}\parallel\alpha^{a'_U})$。他想将 α^{a_V} 替换为 $\alpha^{a'_V}$。然而，这意味着他也必须将签名替换为 $\mathrm{Sig}_V(\mathrm{ID}(U)\parallel\alpha^{a'_V}\parallel\alpha^{a_U})$。不幸的是，因为 W 并不知道 V 的签名算法 Sig_V，所以他无法计算出关于字符串 $\mathrm{ID}(U)\parallel\alpha^{a'_V}\parallel\alpha^{a_U}$ 的 V 的签名。类似地，因为 W 并不知道 U 的签名算法，所以无法用 $\mathrm{Sig}_U(\mathrm{ID}(V)\parallel\alpha^{a'_U}\parallel\alpha^{a_V})$ 来替换 $\mathrm{Sig}_U(\mathrm{ID}(V)\parallel\alpha^{a_U}\parallel\alpha^{a'_V})$。签名的巧妙利用使得 U 和 V 之间可以相互识别。这恰好可以阻止中间入侵攻击。

针对用户知道计算出的会话密钥 K 的情况可以定义几种特性，这里假定在密钥协商协议中 V 是 U 认定的对等方。

(1) 隐式密钥认证。如果 U 可被确保除了 V 之外，没人能计算出 K(特别是敌手应该不能够计算出 K)，则就说该协议提供了隐式密钥认证。

(2) 隐式密钥确认。如果 U 可被确保 V 能计算出 K(假设 V 是按照规定执行了该协议)，并且除了 V 之外，没人能计算出 K，则说该协议提供了隐式密钥确认。

(3) 显式密钥确认。如果 U 可被确保 V 已经计算出了 K，并且除了 V 之外，没人能计算出 K，则说该协议提供了显式密钥确认。

当然，还是希望协议 10.20 可以抵抗所有可能的攻击，而不仅仅是某一种特殊的攻击。已经证明，协议 10.20 是一个认证密钥协商协议，在判定性 Diffie-Hellman 问题难解的假设条件下，该协议为双方都提供了隐式密钥确认。

10.2.6 MTI 密钥协商协议

Matsumoto、Takashima 和 Imai 通过修改 Diffie-Hellman 密钥协商协议构造了一些有趣的密钥协商协议(称之为 MTI 协议)。这些协议不需要用户 U 和 V 之间计算任何签名。因为在该协议的每一个会话中仅执行两个信息传送(一个是从 U 到 V，另一个是从 V 到 U)，因此它们被称为双流(Two-flow)密钥协商协议。相比之下，端-端密钥协商协议是一个 3 段(Three-pass)协议。

本节介绍一种名为 MTI/A0 的 MTI 密钥协商协议[20]。MTI 密钥协商协议是一个比较受关注的协议。在协议提出之后，人们对其安全性做了较多的分析，提出了多个攻击方法，如未知密钥共享攻击[21]、Lim-Lee 攻击[22]、身份冒充攻击[23]等。此外，人们基于 MTI 协议的设计思想，又提出了一些密钥协商协议，如 Goss 协议[24]，但是这些协议大多存在与 MTI 协议类似的缺陷。

协议 10.21 MTI/A0 密钥协商协议。

参数设置：

公开群$(G,\cdot)$和一个阶为 n 的元素 $\alpha\in G$。每一个用户 T 有一个秘密指数 a_T，其中$0\leqslant a_T\leqslant n-1$，对应的公开值为 $b_T=\alpha^{a_T}$，b_T 被包含在 T 的证书中，并被 TA 签名。

具体步骤：

(1) U 随机选取 r_U，$0\leqslant r_U\leqslant n-1$，计算 $s_U=\alpha^{r_U}$，然后 U 将 Cert (U)和 s_U发送给 V。

(2) V 随机选取 r_V，$0\leqslant r_V\leqslant n-1$，计算 $s_V=\alpha^{r_V}$，然后 V 将 Cert(V)和 s_V发送给 U。

(3) V 计算出会话密钥 $K=s_U^{a_V}b_U^{r_V}$，其中他从 Cert (U)中获得了 b_U值；U 计算出会话密钥 $K=s_V^{a_U}b_V^{r_U}$，其中他从 Cert(V)中获得 b_V值。

在会话结束时，U 和 V 都计算出了相同的会话密钥 $K=\alpha^{r_U a_V+r_V a_U}$。

为了便于讨论，将协议的一次会话中被传送的信息绘制成图，如图 10.4 所示。

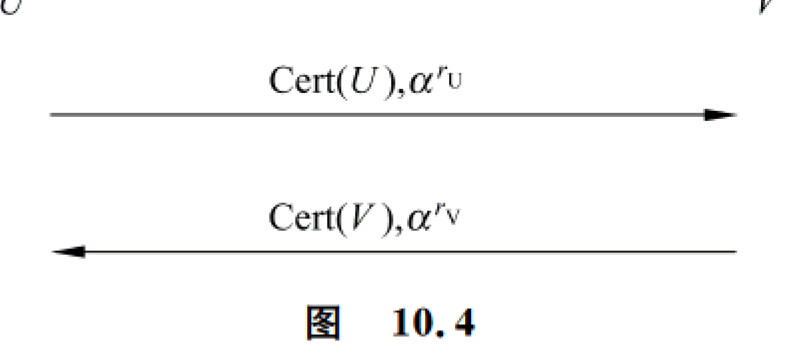


图 10.4

再看一下该协议的安全性。可以很容易地推出，对于一个被动敌手来说，MTI/A0 密钥协商协议和 Diffie-Hellman 密钥协商协议的安全性是一样的。与许多协议一样，在主动敌手的情况下提供可证明安全性是一个难问题。

在协议中如果不使用签名，则可能会遭受中间人侵攻击。事实上，W 确实可能改变 U 和 V 之间传送的值。在图 10.5 中，描述了一种可能会发生的典型场景，它类似于最初提出的应用于 Diffie-Hellman 密钥协商协议的中间人侵攻击。在这种情况下，U 和 V 计算出不同的密钥：U 计算出 $K=\alpha^{r_U a_V + r'_V a_U}$，而 V 计算出 $K=\alpha^{r'_U a_V + r_V a_U}$。然而，$W$ 无法计算由 U 或 V 计算出的密钥（当然这些密钥对他来说是无用的），因为他需要分别知道秘密指数 a_U 和 a_V。

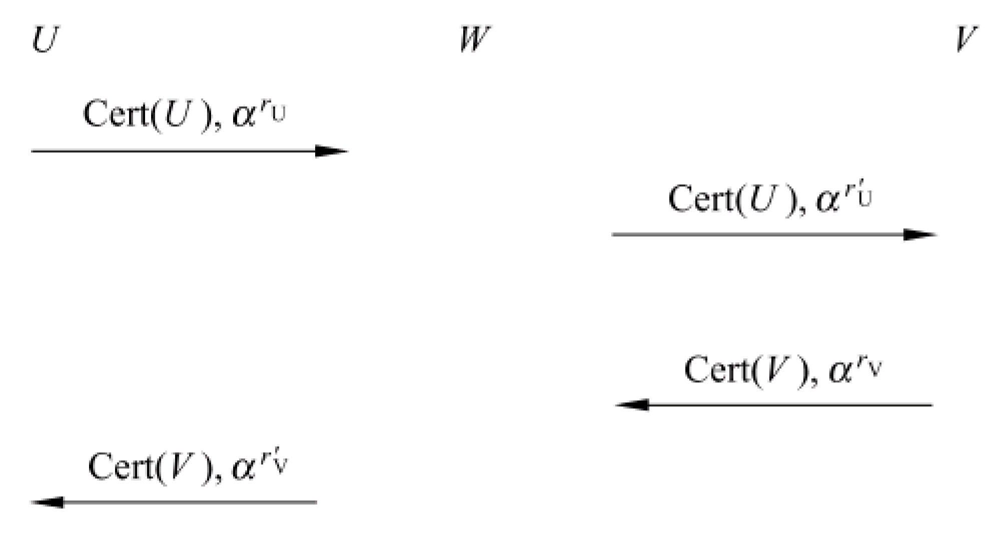


图 10.5　攻击 MTI/A0 的一种典型场景

因此，尽管 U 和 V 已经计算出了不同的密钥，W 却无法计算出其中任何一个密钥，并且他也无法获取关于这些密钥的任何信息（假定 DDH 问题是困难的）。

如果这是关于该协议唯一可能的攻击方法，则说该协议提供了隐式密钥认证。这是因为，尽管存在这种攻击，但 U 和 V 都能保证对方是网络中可以计算出密钥的唯一用户。然而，下面将说明在已知会话密钥攻击模型中会存在由敌手发起的其他攻击。

首先提供一个关于 MTI/A0 并行会话的已知会话密钥攻击。敌手 W 是一个在两个会话中的主动参与者：在会话 S 中，W 假装是和 U 进行对话的 V；在并行发生的会话 S' 中，W 假装是和 V 进行对话的 U。图 10.6 说明了 W 进行的活动。

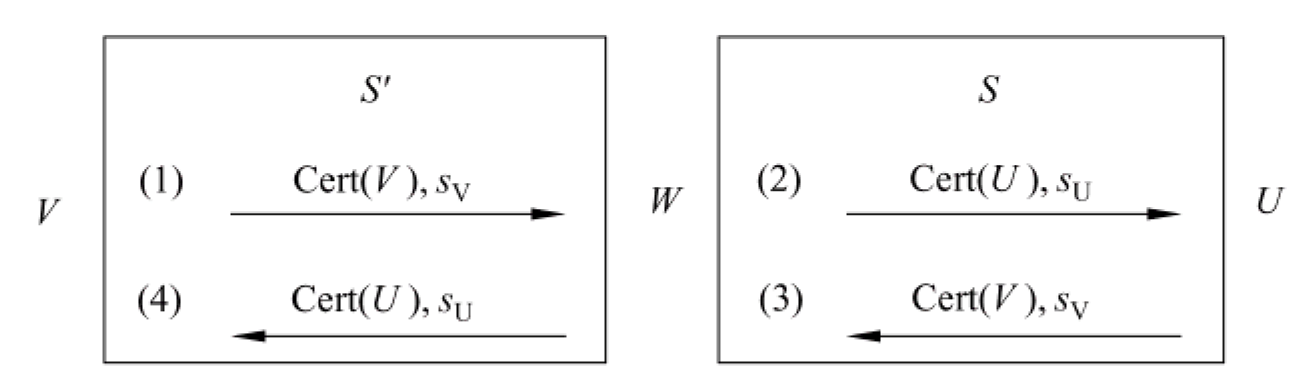


图 10.6　MTI/A0 的已知会话密钥攻击

这两个会话中的流程按照发生的次序进行标号。(1)和(2)分别表示了会话 S' 和 S 的初始信息流。然后 W 将流(1)的信息复制到流(3)中，流(2)的信息复制到流(4)中去。因为这两个会话是并行执行的，所以这是一个并行会话攻击。

当这两个会话结束后，W 请求会话 S' 中的密钥 K，在一个已知会话密钥攻击下，他是被允许这样做的。当然，K 也是会话 S' 的密钥，因此作为一个会话的主动敌手并且没有请求会话密钥的情况下，W 达到了计算密钥的目标。在已知会话密钥攻击模型下，这是一个成功的攻击。

之所以可以进行并行会话攻击，是因为密钥是关于由会话双方提供的输入的一个对称函数值，即 $K=\alpha^{r_U a_V + r_U a_V}=K((r_U, a_U),(r_V, a_V))=K((r_V, a_V),(r_U, a_U))$。

为了消除这种攻击，应该破坏这种对称性。例如，可以使用一个 Hash 函数 h 作为密钥推导函数。假定实际的会话密钥 K 被定义为

$$K = h\left(\alpha^{r_U a_V} \parallel \alpha^{r_V a_U}\right)$$

U(会话的发起者)将计算 $K=h(b_V^{r_U} \parallel s_V^{a_U})$，而 V(会话的响应者)将计算 $K=h(s_U^{a_V} \parallel b_U^{r_V})$。

使用这种修改过的构造会话密钥的方法，前面的攻击将不再有效。这是因为会话 S 和 S'现在有两个不同的密钥：S 的密钥是

$$K_S = h\left(\alpha^{r_U a_V} \parallel \alpha^{r_V a_U}\right)$$

而会话 S'的密钥是

$$K_{S'} = h\left(\alpha^{r_V a_U} \parallel \alpha^{r_U a_V}\right)$$

如果 h 是一个“好”的 Hash 函数(如 h 是一个随机预言)，则 W 在给定 $K_{S'}$的情况下无法计算出 K_S，或在给定 K_S 的情况下无法计算出 $K_{S'}$。

针对 MTI/A0 还有一种已知会话密钥攻击，被称为 Burmester 三角攻击。图 10.7 描述了这种攻击。现在介绍一下三角攻击的基本思想。首先，W 观察到在 U 和 V 之间存在一个会话 S。然后 W 分别参与了 V 和 U 之间的另外两个会话 S_1 和 S_2。在这两个会话中，W 传送由 S 中复制过来的 s_U和 s_V。当然，W 并不知道分别对应于 s_U和 s_V的指数 r_U 和 r_V。然后，当会话 S_1 和 S_2 结束时，W 请求这两个会话的密钥，这在已知会话密钥攻击下是被允许的。

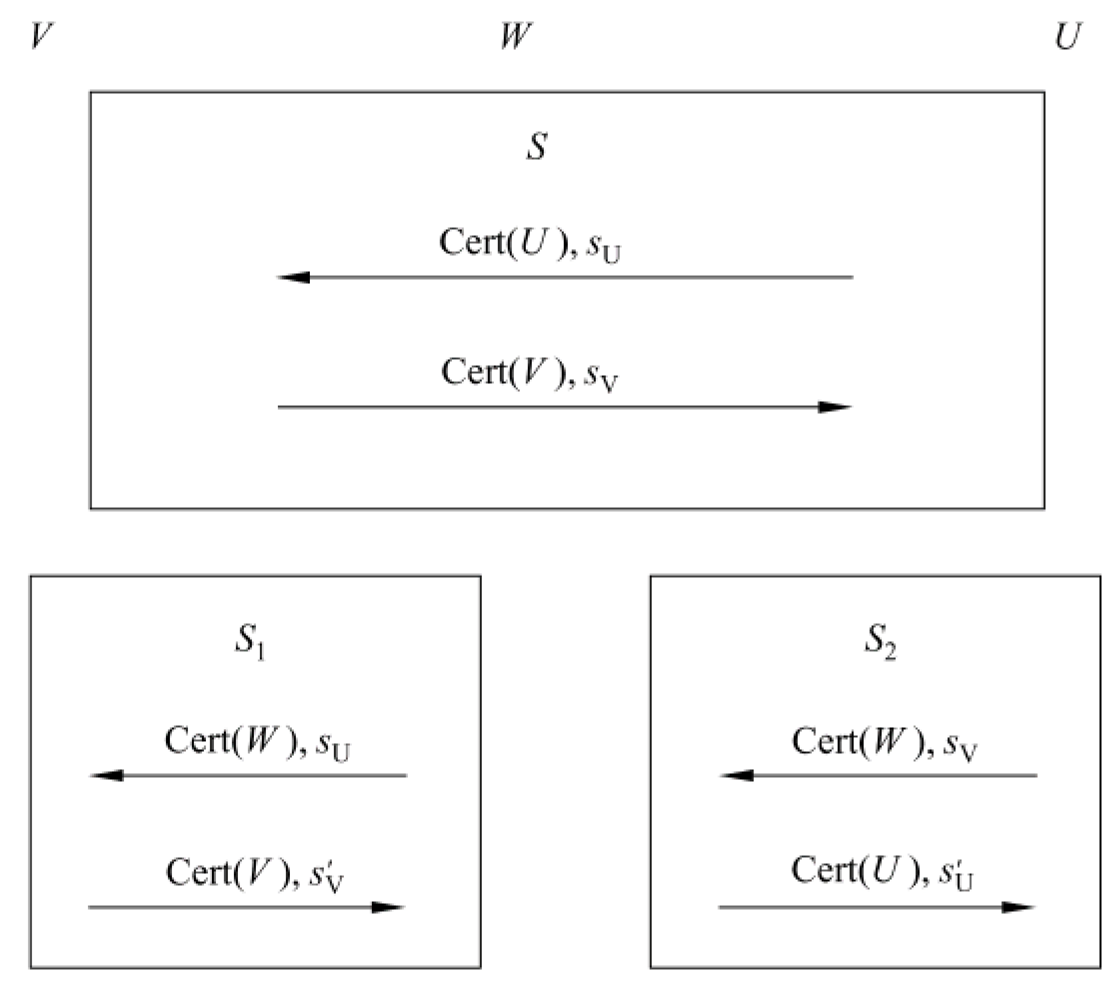


图 10.7 MTI/A0 的 Burmester 三角攻击

会话 S、S_1 和 S_2 的密钥 K、K_1 和 K_2 分别为：$K=\alpha^{r_U a_V + r_V a_U}$、$K_1=\alpha^{r_U a_V + r'_V a_W}$ 和 $K_2=\alpha^{r'_U a_W + r_V a_U}$。

给定 K_1 和 K_2，W 可以按照以下方式计算出 K，即

$$K = \frac{K_1 K_2}{(s'_V s'_U)^{a_W}}$$

因此这是一个成功的已知会话密钥攻击。

通过使用上面描述的密钥推导函数，也可以避免三角攻击。可以猜想 MTI/A0 的修改版对已知会话密钥攻击是安全的。

10.2.7 Girault 密钥协商协议

本小节将描述一种由 Girault 提出的密钥协商协议[25]，该协议不需要证书。在该协议中，公钥值和其拥有者的身份是彼此隐式地相互认证的。

Girault 密钥协商协议结合了 RSA 和基于离散对数的方案的特点。假定 $n=pq$，其中 $p=2p_1+1$，$q=2q_1+1$，p、q、p_1 和 q_1 都是大素数。乘法群 $\mathbb{Z}_n^*$ 与 $\mathbb{Z}_p^* \times \mathbb{Z}_p^*$ 是同构的，因此，$\mathbb{Z}_n^*$ 中任意元素的最大阶是 $p-1$ 和 $q-1$ 的最小公倍数，即为 $2p_1q_1$。令 α 是阶为 $2p_1q_1$ 的群 $\mathbb{Z}_n^*$ 中的一个元素。在 p 和 q 足够大的情况下，由 α 产生的 $\mathbb{Z}_n^*$ 中的循环子群比较适合用来构造离散对数问题。

在 Girault 密钥协商协议中，仅有 TA 知道 n 是如何分解的。n 和 α 是公开参数，而 p、q、p_1 和 q_1 都是秘密的。TA 选取了一个公开的 RSA 加密指数，记为 e。相对应的解密指数 d 是秘密的（通常 $d=e^{-1} \bmod \varphi(n)$）。

每个用户 U 都有一个识别串 ID(U)。由协议 10.22 可知，用户 U 需要从 TA 中获取一个自认证公钥 p_U。U 需要 TA 的帮助来产生公钥 p_U。通过使用公开的信息可以从 p_U 和 ID(U)计算出 $b_U=(p_U^e+\mathrm{ID}(U)) \bmod n$。

协议 10.22 Girault 公钥生成协议。

参数设置：

n 是公开的，d 是仅由 TA 所知道的秘密值。

具体步骤：

(1) U 选取一个秘密指数 a_U，计算 $b_U=\alpha^{a_U} \bmod n$，并将 a_U 和 b_U 发送给 TA。

(2) TA 计算 $p_U=(b_U-\mathrm{ID}(U))^d \bmod n$，并将 p_U 发送给 U。

协议 10.23 描述了 Girault 密钥协商协议。在协议 10.23 中，一次会话中所传送的信息如图 10.8 所示。

图 10.8

在会话结束时，U 和 V 分别计算出密钥

$$K=\alpha^{r_U a_V + r_V a_U} \bmod n = b_V^{r_U} b_U^{r_V} \bmod n$$

协议 10.23 Girault 密钥协商协议。

参数设置：

公开参数 n、e 和 α。

具体步骤：

(1) U 随机选取 r_U，计算 $s_U=\alpha^{r_U} \bmod n$，并将 ID(U)，p_U 和 s_U 发送给 V。

(2) V 随机选取 r_V，计算 $s_V=\alpha^{r_V} \bmod n$，并将 ID(V)，p_V 和 s_V 发送给 U。

(3) U 计算 $K=s_V^{a_U}(p_V^e+\mathrm{ID}(V))^{r_U} \bmod n$，$V$ 计算

$$K=s_U^{a_V}(p_U^e+\mathrm{ID}(U))^{r_V} \bmod n$$

下面分析协议 10.23 是如何抵挡一种特殊类型的攻击。因为 b_U、p_U 和 ID(U)没有被 TA 所签名，所以其他人无法直接验证它们的真实性。假如该信息被冒充 V 的 W 来产生

(它不是和 TA 共同产生的)。W 先产生 ID(U)和一个伪造值 p'_U。他可以很容易计算出 $b'_U=(p'_U)^e+\text{ID}(U)$,但在离散对数问题难解的前提下,他却没办法根据 b'_U 计算出指数 a'_U 来。W 可以假装成 U 来执行一次协议会话,然而,如果不知道 a'_U 的值,则 W 无法计算会话密钥 K。

当 W 想充当一个中间入侵者时,情况也是类似的。W 可以阻止 U 和 V 计算出一个共同的密钥,但是 W 却不能复制 U 或 V 的计算值。因此,就像 MTI/A0 那样,该协议提供了针对这种攻击的隐式密钥认证。

从 Girault 密钥协商协议可以看出,TA 无需 U 给它提供 a_U,它就能直接从 b_U 计算出 p_U。现在要问为什么还要求 U 把值 a_U 提供给 TA 呢?但在这里很重要的一点是 TA 应被确信在为 U 计算 p_U 之前,U 知道 a_U 的值。

为了说明这一点,将描述当 TA 在不先检查用户是否拥有对应于 b_U 的 a_U 值就随便发布公钥 p_U 时,该协议是如何被攻击的。假定 W 选取了一个伪造值 a'_U,并计算出了对应值 $b'_U=\alpha^{a'_U} \bmod n$。

然后计算出对应的公钥值:

$$p'_U=(b'_U-\text{ID}(U))^d \bmod n$$

W 将计算:

$$b'_W=b'_U-\text{ID}(U)+\text{ID}(W)$$

然后将 b'_W 和 ID(W)发送给 TA。假定 TA 发布了公钥值 $p'_W=(b'_W-\text{ID}(W))^d \bmod n$ 给 W。根据 $b'_W-\text{ID}(W)\equiv b'_U-\text{ID}(U) \pmod n$,可以立即推出 $p'_W=p'_U$。

现在假定,在稍后时候,U 和 V 执行了该协议,W 替代了如图 10.9 所示的信息。

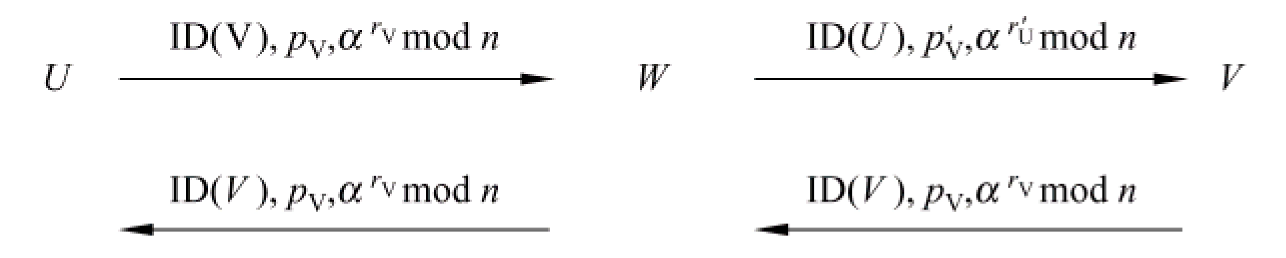


图 10.9 对 Girault 密钥协商协议的攻击

然后 V 计算密钥 $K'=\alpha^{r'_U a_V+r_V a'_U} \bmod n$,同时 U 计算密钥 $K=\alpha^{r_U a_V+r_V a_U} \bmod n$,$W$ 可以计算 $K'=s_V^{a'_U}(p_V^e+\text{ID}(V))^{r'_U} \bmod n$。

因此 W 和 V 共享了一个密钥,而 V 却认为他是在和 U 共享一个密钥。所以 W 可以对由 V 发送到 U 的消息进行解密。

10.2.8 MQV 密钥协商协议

MQV 协议[26]是由 Menezes 等人提出的一个密钥协商协议,这个协议后来由 Law 等人进行了改进[27],并成为 IEEE P1363—2000 标准[28]。在 MQV 协议中,定义了一个特殊的运算:对于 Z_p(p 是素数)上的任意元素 t,定义 $\langle t\rangle=(t \bmod 2^w)+2^w$,即 $\langle t\rangle$ 的结果是一个 w 比特的二进制串(一般取 $w=80$)。

协议 10.24 MQV 密钥协商协议。

参数设置:

系统选择并公布一个适合的大素数 p,并且使得 $p-1$ 的因子中有一个足够大的素数

q,g 是 Z_p 上的一个 q 阶乘法生成元。用户 A 拥有公钥 g^x 和对应的私钥 x,用户 B 拥有公钥 g^y 和对应的私钥 y。

具体步骤:

(1) A 随机选取一个元素 a,$1 \leqslant a \leqslant q-1$,计算 $s_A = g^a \bmod p$,并将 s_A 发送给 B。

(2) B 随机选取一个元素 b,$1 \leqslant b \leqslant q-1$,计算 $s_B = g^b \bmod p$,并将 s_B 发送给 A。

(3) B 接收到消息 s_A,计算 $S_B = b + \langle s_B \rangle y$,然后计算共享密钥 $k = (s_A (g^x)^{\langle s_A \rangle})^{s_B} \bmod p$。

(4) A 接收到消息 s_B,计算 $S_A = a + \langle s_A \rangle x$,然后计算共享密钥 $k = (s_B (g^y)^{\langle s_B \rangle})^{s_A} \bmod p$。

A 和 B 最终的共享密钥为 $k = g^{(a+\langle s_A \rangle x)(b+\langle s_B \rangle y)}$。MQV 密钥协商协议可以提供前向安全性,可以抵抗身份假冒攻击。协议中引入了一个特殊的运算,这个运算破坏了原有的代数结构,从而让人们确信协议的实际安全性。

10.2.9 KEA 密钥协商协议

在 20 世纪 90 年代,美国国家安全局推出了密钥托管机制。在这个机制中,也设计了一个密钥协商协议,称之为 KEA 密钥协商协议[29]。

协议 10.25 KEA 密钥协商协议。

参数设置:

系统选择并公布一个适合的大素数 p,并且使得 $p-1$ 的因子中有一个足够大的素数 q,g 是 Z_p 上的一个 q 阶乘法生成元。用户 A 拥有公钥 g^x 和对应的私钥 x,用户 B 拥有公钥 g^y 和对应的私钥 y。

具体步骤:

(1) A 随机选取一个元素 a,$1 \leqslant a \leqslant q-1$,计算 $s_A = g^a \bmod p$,并将 s_A 发送给 B。

(2) B 随机选取一个元素 b,$1 \leqslant b \leqslant q-1$,计算 $s_B = g^b \bmod p$,并将 s_B 发送给 A。

(3) B 接收到消息 s_A,计算共享密钥 $k = (s_A)^y + (g^x)^b \bmod p$。

(4) A 接收到消息 s_B,计算共享密钥 $k = (s_B)^x + (g^y)^a \bmod p$。

A 和 B 最终的共享密钥为 $k = g^{ay} + g^{bx}$。KEA 密钥协商协议可以视为 MTI 密钥协商协议的一个变体,这个协议不能提供前向安全性。

10.2.10 Internet 密钥协商协议

在基于 Diffie-Hellman 问题的密钥协商协议中,Internet 密钥交换协议也提供了实体认证功能。Internet 密钥交换协议是 IPSec 默认使用的密钥协商协议,它结合了 ISAKMP 协议[30]所提供的认证和密钥交换框架,并使用了部分 Oakley 协议[31]和 SKEME 协议[32]。Internet 密钥交换协议提供了多种可选的工作模式和安全实施方案,所以,这个协议显得比较复杂。在 2005 年,IETF 推出了 Internet 密钥交换协议的第 2 版[33]。

协议 10.26 Internet 密钥交换协议(第 2 版)。

参数设置:

系统选择并公布一个适合的大素数 p,并且使得 $p-1$ 的因子中有一个足够大的素数 q,g 是 Z_p 上的一个 q 阶乘法生成元。用户 A 和用户 B 分别拥有各自的签名私钥以及对应

的验证公钥。

具体步骤：

(1) A 随机选取一个元素 a，$1 \leqslant a \leqslant q-1$，计算 $s_A = g^a \bmod p$，生成消息(CK，$list_A$，s_A，n_A)并将该消息发送给 B，这里，CK 是头格式，$list_A$是 A 支持的密码算法列表，n_A是一个随机数。

(2) B 随机选取一个元素 b，$1 \leqslant b \leqslant q-1$，计算 $s_B = g^b \bmod p$，生成消息(CK，$list_B$，s_B，n_B)并将消息发送给用户 A，这里，CK 是头格式，$list_B$是 B 支持的密码算法列表，n_B是一个随机数。

(3) A 接收到消息后，首先利用两个随机数以及 s_A与 s_B计算一个共享密钥种子，后续消息使用的密钥都源于这个共享密钥种子，然后，A 计算一个认证值 $Auth_A$，最后生成消息(CK，SK(A，$Auth_A$，SA_A，TS_A，TS_B))并将该消息发送给 B，这里，SK 表示将信息加密，A 是用户的标识符，SA_A、TS_A和 TS_B是用于建立一个子安全关联进程的参数。

(4) B 接收到消息后，首先利用两个随机数以及 s_A与 s_B计算一个共享密钥种子，然后 B 计算一个认证值 $Auth_B$，最后，生成消息(CK，SK(B，$Auth_B$，SA_B，TS_A，TS_B))并将消息发送给用户 A，这里，SK 表示将信息加密，B 是用户的标识符，SA_B、TS_A和 TS_B是用于建立一个子安全关联进程的参数。

10.2.11 椭圆曲线上的密钥协商协议

前面介绍的基于公钥密码技术的密钥协商协议，其安全性都是基于有限域上的 Diffie-Hellman 问题。此外，还有一部分基于公钥密码技术的密钥协商协议，其安全性是基于椭圆曲线上的 Diffie-Hellman 问题。本节介绍两个这样的协议。

协议 10.27 Joux 密钥协商协议[34]。

参数设置：

系统选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加法群 G_1的一个生成元 P，并且要求群 G_1和 G_2上的离散对数问题是困难的。系统再选取一个双线性映射 e：$G_1 \times G_1 \rightarrow G_2$。协议有 3 个参与方，用户 A、用户 B 和用户 C。

具体步骤：

(1) A 随机选取一个元素 a，$1 \leqslant a \leqslant q-1$，计算 $s_A = aP$，并将 s_A发送给 B 和 C。

(2) B 随机选取一个元素 b，$1 \leqslant b \leqslant q-1$，计算 $s_B = bP$，并将 s_B发送给 A 和 C。

(3) C 随机选取一个元素 c，$1 \leqslant c \leqslant q-1$，计算 $s_C = cP$，并将 s_C发送给 A 和 B。

(4) A 接收到 B 和 C 发送的消息后，计算共享密钥 $k = e(bP, cP)^a$。

(5) B 接收到 A 和 C 发送的消息后，计算共享密钥 $k = e(aP, cP)^b$。

(6) C 接收到 A 和 B 发送的消息后，计算共享密钥 $k = e(aP, bP)^c$。

与 Diffie-Hellman 密钥协商协议类似，Joux 密钥协商协议没有对发送消息的认证，因此，参与协议的三方都不能保证消息提供者的身份。Joux 密钥协商协议可以抵抗被动攻击，不能抵抗中间人侵攻击等主动攻击。

协议 10.28 Yao-Feng 密钥协商协议[35]。

参数设置：

系统选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加

法群 G_1 的一个生成元 P，并且要求群 G_1 和 G_2 上的离散对数问题是困难的。系统再选取一个双线性映射 $e: G_1 \times G_1 \to G_2$。假定系统中所有用户事先共享一个群 G_1 中的点 Q。

具体步骤：

(1) A 随机选取一个元素 a，$1 \leqslant a \leqslant q-1$，计算 $s_A = aP$，并将 s_A 发送给 B。

(2) B 随机选取一个元素 b，$1 \leqslant b \leqslant q-1$，计算 $s_B = bP$，并将 s_B 发送给 A。

(3) A 接收到 B 发送的消息后，首先计算共享密钥 $k = e(s_B, Q)^a$，然后将 s_A 用共享密钥 k 加密，最后将加密的信息发送给 B。

(4) B 接收到 A 发送的消息后，首先计算共享密钥 $k = e(s_A, Q)^b$，然后将 s_B 用共享密钥 k 加密，最后将加密的信息发送给 A。

(5) A 接收到 B 发送的加密消息后，用共享密钥 k 解密，并验证解密的消息是否与接收的消息 s_B 一致。

(6) B 接收到 A 发送的加密消息后，用共享密钥 k 解密，并验证解密的消息是否与接收的消息 s_A 一致。

在这个协议中，前面两条消息用来建立会话密钥，后面两条消息用来确认会话密钥。

10.3 基于身份的密钥交换协议

Okamoto 协议是第一个基于身份的密钥协商协议[36]。

协议 10.29 Okamoto 基于身份的密钥协商协议。

参数设置：

系统中的可信服务器 TA 选择两个适合的大素数 p 和 q，并计算 $n = pq$，TA 再选取两个 RSA 密码算法参数 e 和 d，即 $ed \equiv 1 \bmod (\varphi(n))$，以及一个与 p 和 q 都互素的数 g，并保证 g 有较高的阶数。TA 公开 n、e 和 g，其他参数保密。

任何用户在使用密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。用户 A 的身份信息为一个模 n 的整数 ID_A，TA 计算用户 A 的私钥为 $(\mathrm{ID}_A)^{-d}$，并将其安全地发送给用户 A；用户 A 的身份信息为一个模 n 的整数 ID_B，TA 计算用户 A 的私钥为 $(\mathrm{ID}_B)^{-d}$，并将其安全地发送给用户 B。

具体步骤：

(1) A 随机选取一个元素 a，$1 \leqslant a \leqslant q-1$，计算 $s_A = (\mathrm{ID}_A)^{-d} g^a$，并将 s_A 发送给 B。

(2) B 随机选取一个元素 b，$1 \leqslant b \leqslant q-1$，计算 $s_B = (\mathrm{ID}_B)^{-d} g^b$，并将 s_B 发送给 A。

(3) A 接收到 B 发送的消息后，计算共享密钥 $k = ((s_B)^e (\mathrm{ID}_B))^a$。

(4) B 接收到 A 发送的消息后，计算共享密钥 $k = ((s_A)^e (\mathrm{ID}_A))^b$。

应用 Okamoto 基于身份的密钥协商协议，A 与 B 最终的共享密钥为 $k = g^{eab}$。由此可知，一个攻击者要想计算 k，他需要知道随机数 a 和 b。Okamoto 基于身份的密钥协商协议使用了参与双方的身份信息，因此可以提供密钥认证。

协议 10.30 Günther 基于身份的密钥协商协议[37]。

参数设置：

系统中的可信服务器 TA 选择一个适合的大素数 p，以及 Z_p 上的一个 q 阶乘法生成元 g。TA 再选取自己的私钥 s 并计算对应的公钥 $g^s \bmod p$。TA 选取一个单向函数 h。

任何用户在使用密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。用户 A 的身份信息是以用户的标识符为变量的单向函数 h 的函数值，即 $ID_A=h(A)$。在注册阶段，用户 A 从 TA 处得到一个 ElGamal 签名(u_A, v_A)：TA 随机选取一个与 $p-1$ 互素的整数 x，$1\leqslant x\leqslant q-1$，计算 $u_A=g^x$，$v_A=(ID_A-su_A)/x$。同样，用户 B 的身份信息为 $ID_B=h(B)$，在注册阶段，用户 B 从 TA 处得到一个 ElGamal 签名(u_B, v_B)：TA 随机选取一个与 $p-1$ 互素的整数 y，$1\leqslant y\leqslant q-1$，计算 $u_B=g^y$，$v_B=(ID_B-su_B)/y$。

具体步骤：

(1) A 将自己的标识符 A 和 ElGamal 签名中的 u_A 一起发送给 B。

(2) B 接收到消息，首先计算 $X_B=g^{H(A)}(g^s)^{-u_A}$，然后将自己的标识符 B 和 ElGamal 签名中的 u_B 一起发送给 A。

(3) A 接收到消息，首先计算 $X_A=g^{H(B)}(g^s)^{-u_B}$，然后 A 随机选取一个元素 a，$1\leqslant a\leqslant q-1$，计算 $s_A=g^a \bmod p$，并将 s_A 发送给 B。

(4) B 接收到消息，然后随机选取一个元素 b，$1\leqslant b\leqslant q-1$，计算 $s_B=g^b \bmod p$，并将 s_B 发送给 A。

(5) A 接收到消息，计算共享密钥 $k=(s_B)^{v_A}(X_A)^a$。

(6) B 计算共享密钥 $k=(s_A)^{v_B}(X_B)^b$。

从理论上讲，Günther 基于身份的密钥协商协议容易遭受类似于那些应用于 MTI 密钥协商协议的已知密钥攻击。

基于数论问题的身份密码体制一直没有得到广泛应用，这是由于有的体制需要防篡改的硬件支持，有的体制用来解析用户私钥的效率低下，有的体制甚至不能抵抗一些用户的合谋攻击等。2001 年，Boneh 和 Franklin 利用双线性映射为数学工具，设计了第一个实用的基于身份的加密体制。这一成果使得基于身份的密码体制的研究和应用取得了突破性进展，这样的进展同样表现在基于身份的密钥协商协议中。

协议 10.31　Smart 基于身份的密钥协商协议[38]。

参数设置：

系统中的可信服务器 TA 选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加法群 G_1 的一个生成元 P，并且要求群 G_1 和 G_2 上的离散对数问题是困难的。系统再选取一个双线性映射 $e: G_1\times G_1\rightarrow G_2$ 和一个将二进制串映射到 G_1 的单向函数 h。TA 再选取自己的私钥 s 并计算对应的公钥 sP。

任何用户在使用 Smart 基于身份的密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。如果一个用户的身份信息为 ID，TA 计算这个用户公钥为 $Q=h(ID)$，私钥为 sQ。

在执行密钥协商协议之前，用户 A 获取的公钥为 Q_A，对应的私钥为 sQ_A；用户 B 获取的公钥为 Q_B，对应的私钥为 sQ_B。

具体步骤：

(1) A 随机选取一个元素 a，$1\leqslant a\leqslant q-1$，计算 $s_A=aP$，并将 s_A 发送给 B。

(2) B 随机选取一个元素 b，$1\leqslant b\leqslant q-1$，计算 $s_B=bP$，并将 s_B 发送给 A。

(3) A 接收到 B 发送的消息后，计算共享密钥 $k=e(aQ_B, (sP))e((sQ_A), s_B)$。

(4) B 接收到 A 发送的消息后，计算共享密钥 $k=e(bQ_A, (sP))e((sQ_B), s_A)$。

应用 Smart 基于身份的密钥协商协议，A 与 B 的共享密钥为 $k=e(aQ_B+bQ_A, (sP))$。

Smart 启发式地分析了协议的安全性，但是 Skim 指出 Smart 基于身份的密钥协商协议并不满足完美的前向安全性[39]。

协议 10.32 Chen-Kudla 基于身份的密钥协商协议[40]。

参数设置：

系统中的可信服务器 TA 选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加法群 G_1 的一个生成元 P，并且要求群 G_1 和 G_2 上的离散对数问题是困难的。系统再选取一个双线性映射 e: $G_1\times G_1\rightarrow G_2$ 和一个将二进制串映射到 G_1 的单向函数 h。TA 再选取自己的私钥 s 并计算对应的公钥 sP。

任何用户在使用 Chen-Kudla 基于身份的密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。如果一个用户的身份信息为 ID，TA 计算这个用户公钥为 $Q=h(\mathrm{ID})$，私钥为 sQ。

在执行密钥协商协议之前，用户 A 获取的公钥为 Q_A，对应的私钥为 sQ_A；用户 B 获取的公钥为 Q_B，对应的私钥为 sQ_B。

具体步骤：

(1) A 随机选取一个元素 a，$1\leqslant a\leqslant q-1$，计算 $s_A=aQ_A$，并将 s_A 发送给 B。

(2) B 随机选取一个元素 b，$1\leqslant b\leqslant q-1$，计算 $s_B=bQ_B$，并将 s_B 发送给 A。

(3) A 接收到 B 发送的消息后，计算共享密钥 $k=e((sQ_A), s_B+aQ_B)$。

(4) B 接收到 A 发送的消息后，计算共享密钥 $k=e((sQ_B), s_A+bQ_A)$。

应用 Chen-Kudla 基于身份的密钥协商协议，A 与 B 的共享密钥为 $k=e(Q_B, Q_A)^{s(a+b)}$。在 Bellare-Rogaway 模型下，如果攻击者不使用 Reveal 询问，Chen 和 Kudla 给出了协议的安全性证明。

协议 10.33 McCullagh-Barreto 基于身份的密钥协商协议[41]。

参数设置：

系统中的可信服务器 TA 选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加法群 G_1 的一个生成元 P，并且要求群 G_1 和 G_2 上的离散对数问题是困难的。系统再选取一个双线性映射 e: $G_1\times G_1\rightarrow G_2$ 和一个将二进制串映射到 $GF(q)$ 的单向函数 h。TA 再选取自己的私钥 s 并计算对应的公钥 sP。

任何用户在使用 McCullagh-Barreto 基于身份的密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。如果一个用户的身份信息为 ID，TA 计算这个用户的公钥为 $Q=h(\mathrm{ID})P+sP$，私钥为 $(h(\mathrm{ID})+s)^{-1}P$。

在执行密钥协商协议之前，用户 A 获取的公钥为 $Q_A=h(\mathrm{ID}_A)P+sP$，对应的私钥为 $S_A=(h(\mathrm{ID}_A)+s)^{-1}P$；用户 B 获取的公钥为 $Q_B=h(\mathrm{ID}_B)P+sP$，对应的私钥为 $S_B=(h(\mathrm{ID}_B)+s)^{-1}P$。

具体步骤：

(1) A 随机选取一个元素 a，$1\leqslant a\leqslant q-1$，计算 $s_A=aQ_B$，并将 s_A 发送给 B。

(2) B 随机选取一个元素 b，$1\leqslant b\leqslant q-1$，计算 $s_B=bQ_A$，并将 s_B 发送给 A。

(3) A 接收到 B 发送的消息后，计算共享密钥 $k=e(s_B, S_A)^a$。

(4) B 接收到 A 发送的消息后，计算共享密钥 $k=e(s_A, S_B)^b$。

应用 McCullagh-Barreto 基于身份的密钥协商协议，A 与 B 的最终共享密钥为$k=e(P,$

$P)^{ab}$。在 Bellare-Rogaway 模型下，如果攻击者不使用 Reveal 询问和单向函数 h 的询问，McCullagh 和 Barreto 给出了协议的安全性证明。后来，Xie 指出了这个协议的一个漏洞[42]，并给出了一个改进方案。然而，Choo 指出原方案和改进方案还是存在安全隐患[43]。

协议 10.34 Wang-Yao-Jiang 基于身份的密钥协商协议[44]。

参数设置：

系统中的可信服务器 TA 选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加法群 G_1 的一个生成元 P，并且要求群 G_1 和 G_2 上的离散对数问题是困难的。系统再选取一个双线性映射 $e: G_1 \times G_1 \rightarrow G_2$、一个将二进制串映射到 G_1 的单向函数 h 和一个 Hash 函数 H。TA 再选取自己的私钥 s 并计算对应的公钥 $R=sP$。

任何用户在使用 Wang-Yao-Jiang 基于身份的密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。如果一个用户的身份信息为 ID，TA 计算这个用户公钥为 $Q=h(\mathrm{ID})$，私钥为 sQ。

在执行密钥协商协议之前，用户 A 获取的公钥为 Q_A，对应的私钥为 sQ_A；用户 B 获取的公钥为 Q_B，对应的私钥为 sQ_B。

具体步骤：

(1) A 随机选取一个元素 a，$1 \leqslant a \leqslant q-1$，计算 $s_A=aP$，$t_A=H(A, B, aP, e(aP, R))+aR$，并将 s_A 和 t_A 发送给 B。

(2) B 随机选取一个元素 b，$1 \leqslant b \leqslant q-1$，计算 $s_B=bP$，$t_B=H(B, A, bP, e(bP, R))+bR$，并将 s_B 和 t_B 发送给 A。

(3) A 接收到 B 发送的消息后，验证 $e(t_B, P)=e(H(B, A, s_B, e(s_B, R))+s_B, R)$ 是否成立。如果等式成立，A 计算共享密钥 $k=H(A, B, as_B, e(s_B, R)^a)$。

(4) B 接收到 A 发送的消息后，验证 $e(t_A, P)=e(H(A, B, s_A, e(s_A, R))+s_A, R)$ 是否成立。如果等式成立，B 计算共享密钥 $k=H(A, B, bs_A, e(s_A, R)^b)$。

应用 Wang-Yao-Jiang 基于身份的密钥协商协议，A 与 B 的最终共享密钥为 $k=H(A, B, abP, e(abP, R))$，这个共享密钥与用户的身份信息无关。文献[44]也给出了协议的安全性证明，并探讨了协议的安全性质。

上面所述的基于身份的密钥协商协议，协议的参与方都是两个。此外，还有参与方为 3 个的基于身份的密钥协商协议，如 Zhang-Liu-Kim 基于身份的密钥协商协议[45]。

协议 10.35 Zhang-Liu-Kim 基于身份的密钥协商协议。

参数设置：

系统中的可信服务器 TA 选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加法群 G_1 的一个生成元 P，并且要求群 G_1 和 G_2 上的离散对数问题是困难的。系统再选取一个双线性映射 $e: G_1 \times G_1 \rightarrow G_2$、一个将二进制串映射到 $GF(q)$ 的单向函数 h 和一个将二进制串映射到 $GF(q)$ 的单向函数。TA 再选取自己的私钥 s 并计算对应的公钥 sP。

任何用户在使用 Zhang-Liu-Kim 基于身份的密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。如果一个用户的身份信息为 ID，TA 计算这个用户公钥为 $Q=h(\mathrm{ID})$，私钥为 sQ。

在执行密钥协商协议之前，用户 A 获取的公钥为 Q_A，对应的私钥为 sQ_A；用户 B 获取

的公钥为 Q_B，对应的私钥为 sQ_B；用户 C 获取的公钥为 Q_C，对应的私钥为 sQ_C。

具体步骤：

(1) A 随机选取一个元素 a，$1 \leqslant a \leqslant q-1$，计算 $s_A = aP$，$t_A = H(aP)(sQ_A) + aP$，并将 s_A 和 t_A 发送给 B 和 C。

(2) B 随机选取一个元素 b，$1 \leqslant b \leqslant q-1$，计算 $s_B = bP$，$t_B = H(bP)(sQ_B) + bP$，并将 s_B 和 t_B 发送给 A 和 C。

(3) C 随机选取一个元素 c，$1 \leqslant c \leqslant q-1$，计算 $s_C = cP$，$t_C = H(cP)(sQ_C) + cP$，并将 s_C 和 t_C 发送给 A 和 B。

(4) A 接收到 B 和 C 发送的消息后，验证 $e(t_B + t_C, P) = e(H(s_B)Q_B + H(s_C)Q_C, sP) e(s_B, s_B)e(s_C, s_C)$。如果等式成立，$A$ 计算共享密钥 $k = e(s_B, s_C)^a$。

(5) B 接收到 A 和 C 发送的消息后，验证 $e(t_A + t_C, P) = e(H(s_A)Q_A + H(s_C)Q_C, sP) e(s_A, s_A)e(s_C, s_C)$。如果等式成立，$B$ 计算共享密钥 $k = e(s_A, s_C)^b$。

(6) C 接收到 A 和 B 发送的消息后，验证 $e(t_A + t_B, P) = e(H(s_A)Q_A + H(s_B)Q_B, sP) e(s_A, s_A)e(s_B, s_B)$。如果等式成立，$C$ 计算共享密钥 $k = e(s_A, s_B)^c$。

10.4 基于口令的密钥协商协议

前面所述的密钥协商协议，如果需要提供认证功能，一般都需要通信双方已经共享了一个长期密钥，比如参与者的公钥或者身份等信息，然后利用长期密钥来生成一个会话密钥。在接下来的通信中，使用会话密钥进行加密信息或者认证消息。但是，这些长期密钥在使用过程中可能并不是很方便，比如需要验证公钥的正确性等。

在实际应用中，人们总是倾向于使用便于记忆的密钥。口令是一种被广泛使用的密钥形式，它通常长度比较短、随机性较差，但是便于记忆和使用。为了方便参与者之间进行认证，人们提出了基于口令的密钥协商协议。基于口令的密钥协商协议允许共享一个口令的两个实体之间完成相互认证，并建立一个会话密钥。

基于口令的密钥协商协议给人的直觉印象是比较难以实现，因为口令包含的信息较少，攻击者容易发起字典攻击。1989 年，Lomas 等人提出了第一个基于口令的协议[46]，在这个协议中，用户和服务器之间共享一个口令，并且要求用户知道服务器的公钥。后来，Bellovin 和 Merritt 提出了加密密钥协商协议的概念[47]，这是第一个严格意义上的基于口令的密钥协商协议。加密密钥协商协议的一般思想是用共享的口令加密需要发送的临时公钥。

协议 10.36 加密密钥协商协议(EKE 协议)。

参数设置：

系统选择并公布一个适合的大素数 p，以及 Z_p 的一个乘法生成元 g，同时给出一个安全参数 L。假定 pw 为用户 A 和用户 B 共享的口令。

具体步骤：

(1) A 随机选取一个元素 a，$1 \leqslant a \leqslant p-2$，计算 $s_A = g^a \bmod p$，并将 s_A 用共享的口令 pw 加密，然后发送给 B。

(2) B 收到消息后，首先随机选取一个元素 b，$1 \leqslant b \leqslant p-2$，计算 $s_B = g^b \bmod p$，并将 s_B 用共享的口令 pw 加密，其次 B 用共享口令 pw 解密收到的消息得到 g^a，计算会话密钥 $k=$

$(g^a)^b$，再其次 B 选取一个随机数 n_B，$1 \leqslant n_B \leqslant 2^L$，并用会话密钥 k 加密，最后，B 将两个加密的消息发送给 A。

(3) A 接收到消息后，首先用共享口令 pw 解密收到的第一部分消息得到 g^b，计算会话密钥 $k=(g^b)^a$，其次 A 用会话密钥解密收到的第二部分消息得到 n_B，再其次 A 选取一个随机数 n_A，$1 \leqslant n_A \leqslant 2^L$，并将 n_A 和 n_B 一起用会话密钥 k 加密，最后用户 A 将加密的消息发送给用户 B。

(4) B 接收到消息后，用会话密钥解密收到的消息得到 n_A 和 n_B，检查 s_B 是否与自己选择的随机数一致，如果一致，B 将 n_A 用会话密钥 k 加密，并将加密的消息发送给 A。

(5) A 接收到消息后，用会话密钥解密收到的消息得到 n_A，检查 s_A 是否与自己选择的随机数一致。

文献[48]讨论了 EKE 协议的安全性。与此同时，文献[49]提出了另一个版本的加密密钥协商协议，称为口令认证密钥协商协议。

协议 10.37　口令认证密钥协商协议（PAK 协议）。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q，q^2 不是 $p-1$ 的因子，g 是 Z_p 上的一个 q 阶乘法生成元。系统再选取 4 个安全 Hash 函数 H_1、H_2、H_3、H_4。

pw 为一个口令，用户 A 拥有口令 pw，用户 B 拥有口令 pw 的影子 $s=(H_1(A, B, \text{pw}))^{(p-1)/q}$。

具体步骤：

(1) A 随机选取一个元素 a，$1 \leqslant a \leqslant q-1$，计算 $s_A = g^a \bmod p$，$m=(H_1(A, B, \text{pw}))^{(p-1)/q} s_A$，并将 m 发送给 B。

(2) B 收到消息 m 后，首先验证 $m \bmod p$ 是否为 0，如果不为 0，B 计算 $s_A = m/s$，其次 B 随机选取一个元素 b，$1 \leqslant b \leqslant q-1$，计算 $s_B = g^b \bmod p$ 和 $k_0=(s_A)^b$，再其次 B 计算 $H_2(A, B, m, s_B, k_0, s)$，最后，$B$ 将 s_B 和计算的 Hash 值发送给 A。

(3) A 收到消息后，首先计算 $k_0=(s_B)^a$，然后计算 $H_2(A, B, m, s_B, k_0, H_1(A, B, \text{pw}))$，并验证计算的 Hash 值与接收的 Hash 值是否一致，如果一致，A 计算共享密钥 $k=H_4(A, B, m, s_B, k_0, H_1(A, B, \text{pw}))$，最后，$A$ 计算 $H_3(A, B, m, s_B, k_0, H_1(A, B, \text{pw}))$，并将计算的 Hash 值发送给 B。

(4) B 收到消息后，首先计算 $H_3(A, B, m, s_B, k_0, s)$，然后验证计算的 Hash 值与接收的 Hash 值是否一致，如果一致，B 计算共享密钥 $k=H_4(A, B, m, s_B, k_0, s)$。

口令认证密钥协商协议一般用于用户和服务器之间的密钥协商，协议中的用户 B 充当服务器角色。在口令认证密钥协商协议中使用了 4 个 Hash 函数，在实际应用中，这 4 个 Hash 函数可以用下面方法得到：将不同的前缀连接在输入数据上，再用标准 Hash 函数 H 作用，例如，$H_i(x)=H(i, x)$。Boyko 等人在提出口令认证密钥协商协议的同时，给出了该协议在 Shoup 模拟模型下的安全性证明。后来，MacKenzie 给出了口令认证密钥协商协议在 Bellare-Rogway 模型下的安全性证明[50]。

Jablon 提出了一个类似于 EKE 协议的口令密钥协商协议，称为 SPEKE 协议[51]。这个密钥协商协议也是以 Diffie-Hellman 密钥协商协议为基础，与 EKE 协议不同的是，在

SPEKE 协议中，口令是用来定义 Diffie-Hellman 密钥协商的基数，而不是指数。

协议 10.38 SPEKE 协议。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $(p-1)/2$ 是一个素数 q，以及一个安全参数 L。系统再选取 3 个安全 Hash 函数 H_1、H_2、H_3。

用户 A 和用户 B 拥有口令 pw。将口令 pw 视为 Z_p 中的元素，计算口令的影子 $P=\text{pw}^2 \bmod p$。

具体步骤：

(1) A 随机选取一个元素 a，$1\leqslant a\leqslant 2^L$，计算 $s_A=P^a \bmod p$，并将 s_A 发送给 B。

(2) B 收到消息 s_A 后，首先随机选取一个元素 b，$1\leqslant b\leqslant 2^L$，计算 $s_B=P^b \bmod p$ 和 Diffie-Hellman 密钥 $k_0=(s_A)^b$，然后 B 验证 k_0 的值是否为 -1、0、1，如果不是，B 计算 $H_1(s_A, s_B, k_0, \text{pw})$，最后 B 将 s_B 和计算的 Hash 值发送给 A。

(3) A 收到消息后，首先计算 Diffie-Hellman 密钥 $k_0=(s_B)^a$，然后计算 $H_1(s_A, s_B, k_0, \text{pw})$，并验证计算的 Hash 值与接收的 Hash 值是否一致，如果一致，A 计算共享密钥 $k=H_3(k_0)$，最后 A 计算 $H_2(s_A, s_B, k_0, \text{pw})$，并将计算的 Hash 值发送给 B。

(4) B 收到消息后，首先计算 $H_2(s_A, s_B, k_0, \text{pw})$，然后验证计算的 Hash 值与接收的 Hash 值是否一致，如果一致，B 计算共享密钥 $k=H_3(k_0)$。

Wu 提出了一个安全远程口令协议[52]，与 EKE 协议相比，该安全远程口令协议效率更高。安全远程口令协议一般用于用户和服务器之间的密钥协商，协议中的用户 B 充当服务器角色。

协议 10.39 安全远程口令协议(SRP 协议)。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $(p-1)/2$ 是一个素数 q。系统再选取一个安全 Hash 函数 H。

pw 为一个口令，用户 A 拥有口令 pw，用户 B 拥有一个随机数 v 和口令 pw 的影子 $s=g^{H(v,\text{pw})}$。

具体步骤：

(1) A 随机选取一个元素 a，$1\leqslant a\leqslant q-1$，计算 $s_A=g^a \bmod p$，并将自己的标识符 A 和 s_A 一起发送给 B。

(2) B 收到 A 发送的消息后，首先随机选取两个元素 b 和 u，$1\leqslant b, u\leqslant q-1$，然后，$B$ 计算中间密钥 $k_0=(s_A s^u)^b$、会话密钥 $k=H(k_0)$ 和数值 $S=s+g^b$，最后，B 将 S、v、u 一起发送给 A。

(3) A 收到消息后，首先利用消息中的 v 计算 $s=g^{H(v,\text{pw})}$，其次 B 计算中间密钥 $k_0=(S-s)^{a+uH(s,\text{pw})}$，会话密钥 $k=H(k_0)$，再其次 B 计算 $m_1=H(A, B, k_0)$，最后 A 将 m_1 的值发送给 B。

(4) B 收到消息后，首先计算 $H(A, B, k_0)$，然后验证计算的 Hash 值与接收的 m_1 值是否一致，如果一致，B 计算 $m_2=H(A, m_1, k_0)$，最后 B 将 m_2 的值发送给 A。

(5) A 收到消息后，首先计算 $H(A, m_1, k_0)$，然后验证计算的 Hash 值与接收的 m_2 值是否一致。

Kwon 提出了 EKE 协议的一个变形，即基于口令认证的密钥协商协议[53]，这个协议融合了其他协议的一些元素。

协议 10.40 基于口令认证的密钥协商协议（AMP 协议）。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q，g 是 Z_p 上的一个 q 阶乘法生成元。系统再选取 4 个安全 Hash 函数 H_1、H_2、H_3、H_4。

pw 为一个口令，用户 A 拥有口令 pw，用户 B 拥有一个随机数 v 和口令 pw 的影子 $s=g^{H_1(v,\mathrm{pw})}$。

具体步骤：

(1) A 随机选取一个元素 a，$1\leqslant a\leqslant q-1$，计算 $s_A=g^a \bmod p$，并将自己标识符 A 和 s_A 一起发送给 B。

(2) B 收到 A 发送的消息后，首先随机选取一个元素 b，$1\leqslant b\leqslant q-1$，然后，$B$ 计算 $T=(s_A s)^b$，$e=H_2(s_A, T, A, B)$，中间密钥 $k_0=(s_A g^e)^b$ 和会话密钥 $k=H_3(k_0)$，最后，B 将 v、T 一起发送给 A。

(3) A 收到消息后，首先利用消息中的 v 计算 $x=H_1(v, \mathrm{pw})$，然后 B 计算 $e=H_2(s_A, T, A, B)$，中间密钥 $k_0=T^{(a+e)/(b+x)}$ 和会话密钥 $k=H_3(k_0)$，最后 A 计算$H_4(s_A, k)$并将得到的 Hash 值发送给 B。

(4) B 收到消息后，首先计算 $H_4(s_A, k)$，然后验证计算的 Hash 值与接收的 Hash 值是否一致，如果一致，B 计算 $H_4(T, k)$。最后 B 将计算的 Hash 值发送给 A。

(5) A 收到消息后，首先计算 $H_4(T, k)$，然后验证计算的 Hash 值与接收的 Hash 值是否一致。

上面介绍的基于口令的密钥协商协议都利用了 Diffie-Hellman 密钥协商协议，此外，还有其他方法来实现基于口令的密钥协商协议。

Bellovin 和 Merritt 建议用口令加密 RSA 算法中的公钥 e，但是，这种方法容易受到攻击。Lucks 提出了开放密钥交换协议（OKE 协议）[54]，在该协议中，RSA 公钥用明文方式发送。后来，MacKenzie 等发现了 RSA 版的 OKE 协议的一个攻击方法[55]，同时，他们提出了一个新的基于 RSA 的 OKE 协议，称为用口令信息实现网络安全认证协议（简称 SNAPI 协议）。

协议 10.41 SNAPI 协议。

参数设置：

系统选择两个安全参数 L_1 和 L_2。

系统选取一个 Hash 函数 H，它的输出长度 $l\geqslant L_1+L_2$。系统再选取 3 个 Hash 函数 H_1、H_2 和 H_3，它们的输出长度为 L_2。定义集合 $X_N=\{x: x\leqslant 2^l-(2^l \bmod N)\wedge(x, N)=1\}$。

用户 A 和用户 B 拥有一个共享口令 pw。

具体步骤：

(1) A 随机选取一个长为 L_2 的 0、1 符号串 a，以及 RSA 公钥 N 和 e，$2^{L_1-2}\leqslant N\leqslant 2^{L_1}$，$2^{L_1}\leqslant e\leqslant 2^{L_1+1}$，$A$ 将自己的标识符连同随机符号串 a、公钥 N 和 e 一起发送给 B。

(2) B 收到消息后，首先检查 N 和 e 是否在规定的范围内，e 是否是一个素数，以及 a 的

长度是否为 L_2，如果满足条件，B 随机选取一个长为 L_2 的 0、1 符号串 b，然后，B 计算 $p=H(N, e, a, b, A, B, \mathrm{pw})$，并随机选取一个与 N 互素的整数 f，如果 p 不属于集合 X_N，令 $q=f$，否则，计算 $q=pf^e \bmod N$，最后，B 将 b 和 q 一起发送给 A。

(3) A 收到消息后，首先检查 b 的长度是否为 L_2，以及 q 和 N 是否互素，如果条件满足，B 计算 $p=H(N, e, a, b, A, B, \mathrm{pw})$，并检查 p 是否属于集合 X_N，如果不属于，协议终止，如果属于，计算 $f=(q/p)^d \bmod N$，然后 A 计算 $r=H_1(N, e, a, b, A, B, q, f)$，最后，$A$ 将 r 的值发送给 B。

(4) B 收到消息后，首先检查 q 是否属于集合 X_N，如果不属于则协议终止，如果属于则计算 $r=H_1(N, e, a, b, A, B, q, f)$，然后 B 验证计算的 r 的值与得到的 Hash 值是否一致，如果通过验证，B 计算 $t=H_2(N, e, a, b, A, B, q, f)$，以及会话密钥 $k=H_3(N, e, a, b, A, B, q, f)$，最后 B 将 t 的值发送给 A。

(5) A 收到消息后，首先计算 $t=H_2(N, e, a, b, A, B, q, f)$，然后 A 验证计算的 t 值与接收的 Hash 值是否一致，如果一致，A 计算会话密钥 $k=H_3(N, e, a, b, A, B, q, f)$。

除了使用 RSA 加密算法来实现基于口令的密钥协商协议之外，Lee 等还提出了一个利用 Nyberg-Rueppel 协议实现的基于口令的密钥协商协议[56]。

协议 10.42 Lee-Sohn-Yang-Won 基于口令的密钥协商协议。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q，g 是 Z_p 上的一个 q 阶乘法生成元。系统再选取两个安全的 Hash 函数 H_1、H_2。

pw 为一个口令，用户 A 拥有口令 pw，用户 B 拥有口令 pw 的影子 $s=g^{-H_2(\mathrm{pw})}$。

具体步骤：

(1) A 首先随机选取两个元素 a 和 r，$1\leqslant a, r\leqslant q-1$，然后 A 计算 $e=g^{a-r} \bmod p$，$f=r+eH_2(\mathrm{pw}) \bmod q$，最后 A 将自己的标识符 A 连同 e 和 f 一起发送给 B。

(2) B 收到消息后，首先计算 $s_A=s^e g^f e$，然后 B 随机选取一个元素 b，$1\leqslant b\leqslant q-1$，计算 $s_B=g^b \bmod p$，共享密钥 $k=(s_A)^b$，最后 B 计算 $H_1(k, e, f)$，并将 s_B 和计算的 Hash 值一起发送给 A。

(3) A 收到消息后，首先计算共享密钥 $k=(s_B)^a$，然后 A 计算 $H_1(k, e, f)$，并验证计算的 Hash 值与接收的 Hash 值是否一致，如果一致，A 计算 $H_1(k, s_B)$，最后 A 将计算的 Hash 值发送给 B。

(4) B 收到消息后，计算 $H_1(k, s_B)$，并且验证计算的 Hash 值与接收的 Hash 值是否一致。

在 Lee-Sohn-Yang-Won 基于口令的密钥协商协议中，用户 A 所需要的计算资源比较少。这个协议可以抵抗被动攻击，不能抵抗主动攻击。一个攻击者 C 可以随机选择一个数 c，计算 $e=g^c \bmod p$，再随机选择一个数 f，并将 A、e、f 一起发送给用户 B。这样，攻击者 C 知道用户 B 计算的共享密钥 $k=(s_B)^{-H_2(\mathrm{pw})e+f+c}$。在收到用户 B 发送的消息 s_B，$H_1(k, e, f)$后，攻击者可以猜测一个口令 w，计算 $k_1=(s_B)^{-H_2(w)e+f+c}$，$h=H_1(k_1, e, f)$，通过比较 h 的值与收到的 Hash 值，攻击者可以验证猜测的口令是否正确。

上面介绍了几个基于口令的密钥协商协议，下面介绍一个基于口令的密钥分配协议，即 GLNS 秘密公钥协议。在 GLNS 秘密公钥协议中，有两个用户和一个服务器，每个用户与

服务器之间共享一个秘密口令，最终服务器为两个用户选择一个会话密钥，并且安全地发送给两个用户。

协议 10.43　GLNS 秘密公钥协议[57]。

参数设置：

协议包括两个用户 A、B 和一个服务器 S。系统选取两个安全的 Hash 函数 H_1、H_2。

用户 A 和服务器 S 共享一个口令 pwa，用户 B 和服务器 S 共享一个口令 pwb。

具体步骤：

(1) 如果 A 想要和 B 建立会话密钥，他将 A 和 B 的标识符一起发送给 S。

(2) S 收到 A 发送的消息后，首先选取一个随机数 n_S，两个临时加密公钥 K_S 和 K'_S，然后将 K_S 用与 A 共享的密钥 pwa 加密，将 K'_S 用与 B 共享的密钥 pwb 加密，最后，S 将 A 和 B 的标识符、选取的随机数 n_S 和两个加密的临时加密公钥一起发送给 A。

(3) A 收到消息后，首先利用他与 S 共享的密钥 pwa 解密第一个加密了的临时公钥，得到 K_S，其次，A 选取 4 个随机数 n_A、n'_A、c_A、r_A，再其次，A 将从 S 处得到的随机数 n_S 用他与 S 共享的密钥 pwa 加密，再用得到的临时加密公钥 K_S 加密消息 A、B、n_A、n'_A、c_A、$E_{pwa}(n_S)$，最后 A 将加密的消息以及 n_S、r_A、$E_{pwb}(K'_S)$ 一起发送给 B。

(4) B 收到消息后，首先利用他与 S 共享的密钥 pwb 解密加密了的临时公钥，得到 K'_S，其次，B 选取 4 个随机数 n_B、n'_B、c_B、r_B，再其次，B 将从 A 转发的随机数 n_S 用他与 S 共享的密钥 pwb 加密，再用得到的临时加密公钥 K'_S 加密消息 B、A、n_B、n'_B、c_B、$E_{pwb}(n_S)$，最后 A 将加密的消息连同从 A 处得到的第一个加密消息一起发送给 S。

(5) S 收到消息后，首先用临时公钥 K_S 对应的私钥解密第一个加密的消息，得到 A、B、n_A、n'_A、c_A、$E_{pwa}(n_S)$，再用他与 A 共享的密钥 pwa 解密 $E_{pwa}(n_S)$，得到 n_S，并验证得到的 n_S 是否与自己选取的 n_S 一致，然后用临时公钥 K'_S 对应的私钥解密第二个加密的消息，得到 B、A、n_B、n'_B、c_B、$E_{pwb}(n_S)$，再用他与 B 共享的密钥 pwb 解密 $E_{pwb}(n_S)$，得到 n_S，并验证得到的 n_S 是否与自己选取的 n_S 一致，如果两次验证都通过，S 随机选取一个会话密钥 k，用他与 A 共享的密钥 pwa 加密消息 n_A 和 $k \oplus n'_A$，用他与 B 共享的密钥 pwb 加密消息 n_B 和 $k \oplus n'_B$，最后 S 将两个加密的消息发送给 B。

(6) B 收到消息后，首先用他与 S 共享的密钥 pwb 解密后面一个加密的消息，得到 n_B 和 $k \oplus n'_B$，其次 B 验证 n_B 是否与自己选取的 n_B 一致，如果一致，B 将 $k \oplus n'_B$ 与自己选取的 n'_B 做异或运算，得到会话密钥 k，再其次，B 计算 $H_1(r_A)$，并将得到的 Hash 值与 r_B 一起用会话密钥 k 加密，最后，B 将从 S 处得到的前面一个加密消息连同用会话密钥 k 加密的消息一起发送给 A。

(7) A 收到消息后，首先用他与 S 共享的密钥 pwa 解密前面一个加密的消息，得到 n_A 和 $k \oplus n'_A$，并且验证 n_A 是否与自己选取的 n_A 一致，如果一致，A 将 $k \oplus n'_A$ 与自己选取的 n'_A 做异或运算，得到会话密钥 k，其次，A 用会话密钥 k 解密后面一个加密的消息，得到 $H_1(r_A)$ 和 r_B，再其次，A 计算 $H_1(r_A)$，并验证结果与得到 Hash 值是否一致，如果一致，A 计算 $H_2(r_B)$，并用会话密钥 k 加密，最后 A 将加密的结果发送给 B。

(8) B 收到消息后，首先用会话密钥 k 解密消息，得到 $H_2(r_B)$，然后 B 计算 $H_2(r_B)$，最后 B 验证计算的 Hash 值与接收的 Hash 值是否一致。

在 GLNS 秘密公钥协议中，用户 A 选取了 4 个随机数 n_A、n'_A、c_A、r_A。在这 4 个随机数

中，n_A是用来与服务器完成认证的，并保证密钥的新鲜性，n'_A是用来掩饰会话密钥 k，c_A是用来随机化第(3)步中的加密消息的，r_A是用来与用户 B 认证。同样，可分析用户 B 选取的4个随机数的作用。由于GLNS秘密公钥协议比较复杂，后来人们提出了一些简化的方案[58,59]。

随着计算机网络的不断发展，需要为在不同服务器下的用户建立会话密钥。为此，在2002年，Byun等人提出了一个跨域密钥协商协议，用户到用户的口令认证密钥交换协议[60]。

协议 10.44 用户到用户的口令认证密钥交换协议(C2C-PAKE协议)。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q，g 是 Z_p 上的一个 q 阶乘法生成元。系统再选取5个安全的Hash函数 H_1、H_2、H_3、H_4和 H_5。E 是一个对称密码算法。

用户 A 和服务器SA共享一个口令pwa，用户 B 和服务器SB共享一个口令pwb，服务器SA和服务器SB共享一个长期会话密钥 K。

具体步骤：

(1) 如果 A 想要和 B 建立会话密钥，他随机选取一个元素 x，$1\leqslant x\leqslant q-1$，计算 $E_{\text{pwa}}(g^x)$，并将 A 和 B 的标识符连同加密信息一起发送给SA。

(2) SA收到 A 发送的消息后，首先用它与 A 共享的密钥pwa解密 $E_{\text{pwa}}(g^x)$，并且随机选取两个元素 y 和 r，$1\leqslant y$，$r\leqslant q-1$，计算 $E_{\text{pwa}}(g^y)$ 和 $g^{\text{pwa}\cdot r}$，其次，SA产生一个票据 $\text{Ticket}_B=E_K(g^{\text{pwa}\cdot r}, g^r, A, B, L)$，其中 L 是票据的生存时间，再其次，SA计算 $R=H_1(g^{xy})$，$E_R(g^x\oplus g^r, A, B)$，最后，SA将 $E_R(g^x\oplus g^r, A, B)$、$E_{\text{pwa}}(g^y)$、Ticket_B和 L 一起发送给 A。

(3) A 收到消息后，首先利用pwa解密 $E_{\text{pwa}}(g^y)$得到 g^y，然后 A 计算 $R=H_1(g^{xy})$，并用 R 解密 $E_R(g^x\oplus g^r, A, B)$，从而得到 g^r，最后 A 将 Ticket_B、A、L 一起发送给 B。

(4) B 收到消息后，首先随机选取一个元素 x'，$1\leqslant x'\leqslant q-1$，然后，$B$ 计算$E_{\text{pwb}}(g^{x'})$，最后 B 将 Ticket_B、$E_{\text{pwb}}(g^{x'})$、A、B、L 一起发送给SB。

(5) SB收到消息后，首先用密钥 K 解密 Ticket_B得到 $g^{\text{pwa}\cdot r}$，并计算 $h=H_4(g^{\text{pwa}\cdot r})$，其次，SB随机选取一个元素 r'，$1\leqslant r'\leqslant q-1$，计算 $g^{\text{pwb}\cdot r\cdot r'}$，再随机选取一个元素 y'，$1\leqslant y'\leqslant q-1$，计算 $R'=H_2(g^{x'y'})$ 和 $E_{\text{pwb}}(g^{y'})$，再其次，SB计算 $E_{R'}(g^{\text{pwa}\cdot r\cdot r'}\oplus g^{x'}, A, B)$和 $E_h(g^{\text{pwb}\cdot r\cdot r'})$，最后SB将 $E_{R'}(g^{\text{pwa}\cdot r\cdot r'}\oplus g^{x'}, A, B)$、$E_{\text{pwb}}(g^{y'})$、$E_h(g^{\text{pwb}\cdot r\cdot r'})$一起发送给 B。

(6) B 收到消息后，首先用他与SB共享的密钥pwb解密 $E_{\text{pwb}}(g^{y'})$得到 $g^{y'}$，其次，B 计算 $R'=H_2(g^{x'y'})$，用 R'解密消息 $E_{R'}(g^{\text{pwa}\cdot r\cdot r'}\oplus g^{x'}, A, B)$，得到 $g^{\text{pwa}\cdot r\cdot r'}$，从而计算 $\text{cs}=H_5(g^{\text{pwa}\cdot\text{pwb}\cdot r\cdot r'})$，再其次，$B$ 随机选取一个元素 a，$1\leqslant a\leqslant q-1$，计算 $E_{\text{cs}}(g^a)$，最后 B 将 $E_{\text{cs}}(g^a)$、$E_h(g^{\text{pwb}\cdot r\cdot r'})$一起发送给 A。

(7) A 收到消息后，首先由 g^r 和pwa计算 $h=H_4(g^{\text{pwa}\cdot r})$，并用 h 解密消息 $E_h(g^{\text{pwb}\cdot r\cdot r'})$，得到 $g^{\text{pwb}\cdot r\cdot r'}$，其次 A 计算 $\text{cs}=H_5(g^{\text{pwa}\cdot\text{pwb}\cdot r\cdot r'})$，并用cs解密 $E_{\text{cs}}(g^a)$得到 g^a，再其次，A 随机选取一个元素 b，$1\leqslant b\leqslant q-1$，计算 $E_{\text{cs}}(g^a)$和会话密钥 $k=H_3(g^{ab})$，最后，B 计算 $E_k(g^a)$，并将 $E_k(g^a)$、$E_{\text{cs}}(g^b)$一起发送给 B。

(8) B 收到消息后，首先用 cs 解密消息 $E_{cs}(g^b)$ 得到 g^b，其次，B 计算会话密钥$k=H_3(g^{ab})$，再其次，B 用会话密钥 k 解密消息 $E_k(g^a)$，得到 g^a，并验证 g^a 是否与自己选择的 g^a 一致，如果一致，B 计算 $E_k(g^b)$，最后 B 将 $E_k(g^b)$ 发送给 A。

(9) A 收到消息后，用会话密钥 k 解密消息 $E_k(g^b)$，得到 g^b，并验证 g^b是否与自己选择的 g^a一致。

在 Byun 等人提出用户到用户的口令认证密钥交换协议时，启发式地讨论了协议的安全性。在此之后，人们对这个协议的安全性做了进一步的分析，发现了一些对该协议的攻击方法，如口令攻击[61]、Wang-Wang-Xu 攻击[62]、内部攻击[63]等，人们提出了一些改进方案，但是还是存在一些安全漏洞。针对现有的攻击方法，Yao、Feng 和 Han 提出了一个改进方案[64]。

协议 10.45 IC2C-PAKE 协议。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q。系统选取一个将二进制串映射到 Z_q 的单向函数 H，并且可以找到一个 h(最小的正整数)，使得 $H(\mathrm{pw}, \mathrm{ID}, h)$是 Z_p 上的一个 q 阶乘法生成元[65]。系统再选取 4 个安全的 Hash 函数 H_1、H_2、H_3 和 H_4。E 是一个对称密码算法。

用户 A 和服务器 SA 共享一个口令 pwa，用户 B 和服务器 SB 共享一个口令 pwb，服务器 SA 和服务器 SB 共享一个长期会话密钥 K。

具体步骤：

(1) 如果 A 想要和 B 建立会话密钥，他随机选取两个元素 x, x'，$1\leqslant x, x'\leqslant q-1$，计算 $g_A=H(\mathrm{pwa}, A, h_A) \bmod p$ 和 $g=H(A, B, h) \bmod p$，并将 A 和 B 的标识符连同 g^x、$(g_A)^{x'}$一起发送给 SA。

(2) SA 收到 A 发送的消息后，首先利用 pwa 计算 $g_A=H(\mathrm{pwa}, A, h_A) \bmod p$，并且随机选取两个元素 s 和 s'，$1\leqslant s, s'\leqslant q-1$ 以及一个加密密钥 sk，计算$(g^x)^s$和$((g_A)^{x'})^{s'}$，其次，SA 产生一个票据 $\mathrm{Ticket}_B=E_K((g^x)^s, s, k, A, B, L)$，其中 L 是票据的生存期，再其次，SA 计算 $R=H_1(((g_A)^{x'})^{s'})$，$E_R(\mathrm{sk}, A, B)$，最后 SA 将$E_R(\mathrm{sk}, A, B)$、$(g_A)^{s'}$、Ticket_B、L 一起发送给 A。

(3) A 收到消息后，首先利用$(g_A)^{s'}$计算 $R=H_1(((g_A)^{s'})^{x'})$，然后 A 用R 解密 $E_R(\mathrm{sk}, A, B)$，从而得到 sk，最后 A 将 A、B、Ticket_B、L 一起发送给 B。

(4) B 收到消息后，首先随机选取两个元素 y、y'，$1\leqslant y, y'\leqslant q-1$，然后 B 计算 $g_B=H(\mathrm{pwb}, B, h_B) \bmod p$ 和 $g=H(A, B, h) \bmod p$，最后 B 将 A、B、g^y、$(g_B)^{y'}$、Ticket_B、L 一起发送给 SB。

(5) SB 收到消息后，首先用密钥 K 解密 Ticket_B 得到 s、$g^{x\cdot s}$和 sk，并且计算 $g_B=H(\mathrm{pwb}, B, h_B) \bmod p$，其次 SB 随机选取两个元素 t、t'，$1\leqslant t, t'\leqslant q-1$，并且计算$(g^{x\cdot s})^t$、$(g^y)^{s\cdot t}$和$((g_B)^{y'})^{t'}$，再其次，SB 计算 $R'=H_2(((g_B)^{y'})^{t'})$，以及 $E_{R'}(g^{x\cdot s\cdot t}, A, B)$和 $E_{sk}(g^{y\cdot s\cdot t})$，最后，SB 将 $E_{R'}(g^{x\cdot s\cdot t}, A, B)$、$(g_B)^{t'}$、$E_{sk}(g^{y\cdot s\cdot t})$一起发送给 B。

(6) B 收到消息后，首先利用$(g_B)^{t'}$计算 $R'=H_2(((g_B)^{t'})^{y'})$，其次 B 用 R'解密消息 $E_{R'}(g^{x\cdot s\cdot t}, A, B)$，得到 $g^{x\cdot s\cdot t}$，从而计算 $\mathrm{cs}=H_3((g^{x\cdot s\cdot t})^y)$，再其次，$B$ 随机选取一个元素 a，$1\leqslant a\leqslant q-1$，计算 $E_{cs}(g^a)$，最后 B 将 $E_{cs}(g^a)$和 $E_{sk}(g^{y\cdot s\cdot t})$一起发送给 A。

(7) A 收到消息后，首先用 k 解密消息 $E_{sk}(g^{y \cdot s \cdot t})$，得到 $g^{y \cdot s \cdot t}$，其次，A 计算 $cs = H_3((g^{y \cdot s \cdot t})^x)$，再其次，$A$ 随机选取一个元素 b，$1 \leqslant b \leqslant q-1$，计算 $E_{cs}(g^b)$ 和会话密钥 $k = H_4(g^{ab})$，并用 cs 解密 $E_{cs}(g^a)$ 得到 g^a，最后，B 计算 $E_k(g^a)$，并将 $E_k(g^a)$，$E_{cs}(g^b)$ 一起发送给 B。

(8) B 收到消息后，首先用 cs 解密消息 $E_{cs}(g^b)$ 得到 g^b，其次，B 计算会话密钥 $k = H_4(g^{ab})$，再其次，B 用会话密钥 k 解密消息 $E_k(g^a)$，得到 g^a，并验证 g^a 是否与自己选择的 g^a 一致，如果一致，B 计算 $E_k(g^b)$，最后 B 将 $E_k(g^b)$ 发送给 A。

(9) A 收到消息后，用会话密钥 k 解密消息 $E_k(g^b)$，得到 g^b，并验证 g^b 是否与自己选择的 g^b 一致。

基于口令的密钥协商协议是一类很实用的密钥交换协议，有着广阔的应用前景，但其理论基础还不够完善，针对这个问题，我们对基于口令的密钥协商协议的设计与分析问题进行了系统研究，并设计了一些可证明安全的基于口令的密钥协商协议，这些工作已在 3.5.2 小节中做了介绍，这里不再赘述。

10.5 群组密钥协商协议

群组密钥协商协议是一种网络中两个或更多用户组成的一个子集可以构建一个共享密钥（如组密钥）的密钥协商协议，也称会议密钥协商协议。本节假定有 n 个参与者 U_1，U_2，…，U_n，如果 $i \equiv j \bmod n$，则认为 U_i 和 U_j 是同一个参与者。

在 Diffie-Hellman 提出第一个两方密钥协商协议之后，Ingemarsson 等人在 1982 年将这个两方密钥协商协议推广到多方密钥协商协议的情形[66]。在推广的密钥协商协议中，所有的参与者组成一个环，第 i 个参与者从第 $i-1$ 个参与者处接收信息，并向第 $i+1$ 个参与者发送信息。

协议 10.46 Ingemarsson-Tang-Wong 群组密钥协商协议。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q，g 是 Z_p 上的一个 q 阶乘法生成元。

具体步骤：

(1) 在第 1 轮中，每个用户 $U_i(1 \leqslant i \leqslant n)$ 随机选取一个元素 r_i，$1 \leqslant r_i \leqslant q-1$，计算 $g^{r_i} \bmod p$，并将 g^{r_i} 发送给 U_{i+1}。

(2) 在第 k 轮 $(2 \leqslant k \leqslant n-1)$ 中，用户 U_i 用在上一轮中从用户 U_{i-1} 处得到的消息 $g^{r_{i-1} r_{i-2} \cdots r_{i-k+1}}$ 以及自己选取的随机数 r_i，计算 $g^{r_i r_{i-1} r_{i-2} \cdots r_{i-k+1}} \bmod p$，并将 $g^{r_i r_{i-1} r_{i-2} \cdots r_{i-k+1}} \bmod p$ 发送给用户 U_{i+1}。

(3) 在 $n-1$ 轮后，所有的用户可以计算共享的群组密钥 $k = g^{r_1 r_2 \cdots r_n}$。

在安全性方面，Ingemarsson-Tang-Wong 群组密钥协商协议可以抵抗被动攻击。在效率方面，Ingemarsson-Tang-Wong 群组密钥协商协议需要 $n-1$ 轮，每个用户需要发送 $n-1$ 条消息，需要做 n 次幂运算。

Steiner 等人提出了 3 个群组密钥协商协议[67]，分别称为 GDH.1、GDH.2 和 GDH.3。这 3 个群组密钥协商协议可以视为 Ingemarsson-Tang-Wong 群组密钥协商协议的变形，它

们最终得到的共享密钥都是 $k=g^{r_1 r_2 \cdots r_n}$。

协议 10.47　GDH. 3 群组密钥协商协议。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q，g 是 Z_p 上的一个 q 阶乘法生成元。

具体步骤：

(1) 在第 1 轮中，用户 U_1 随机选取一个元素 r_1，$1\leqslant r_1\leqslant q-1$，计算 $g^{r_1} \bmod p$，并将 $g^{r_1} \bmod p$ 发送给用户 U_2。

(2) 在第 j 轮($2\leqslant j\leqslant n-2$)中，用户 U_j 随机选取一个元素 r_j，$1\leqslant r_j\leqslant q-1$，利用在上一轮中从用户 U_{j-1} 处得到的消息 $g^{r_1 r_2 \cdots r_{i-1}}$ 及随机数 r_j，计算 $g^{r_1 r_2 \cdots r_{i-1} r_i} \bmod p$，并将 $g^{r_1 r_2 \cdots r_{i-1} r_i} \bmod p$ 发送给用户 U_{j+1}。

(3) 在 $n-1$ 轮中，用户 U_{m-1} 随机选取一个元素 r_{m-1}，$1\leqslant r_{m-1}\leqslant q-1$，利用在上一轮中从用户 U_{m-2} 处得到的消息 $g^{r_1 r_2 \cdots r_{m-2}}$ 及随机数 r_{m-1}，计算 $g^{r_1 r_2 \cdots r_{m-2} r_{m-1}} \bmod p$，并将 $g^{r_1 r_2 \cdots r_{m-2} r_{m-1}} \bmod p$ 广播给协议所有的参与者。

(4) 在第 n 轮中，用户 U_i($1\leqslant i\leqslant n-1$)在收到广播消息后，计算随机数 r_i 模 $p-1$ 的逆 f_i，利用在上一轮中从用户 U_{m-1} 处得到的消息 $g^{r_1 r_2 \cdots r_{m-2} r_{m-1}}$ 及 f_i，计算 $(g^{r_1 r_2 \cdots r_{m-2} r_{m-1}})^{f_i} \bmod p$，并将 $(g^{r_1 r_2 \cdots r_{m-2} r_{m-1}})^{f_i} \bmod p$ 发送给用户 U_m。

(5) 在第 $n+1$ 轮中，用户 U_m 首先随机选取一个元素 r_m，$1\leqslant r_m\leqslant q-1$，利用在第 $n-1$ 轮中从用户 U_{m-1} 处得到的消息 $g^{r_1 r_2 \cdots r_{m-2} r_{m-1}}$ 及随机数 r_m，计算共享的群组密钥 $k=g^{r_1 r_2 \cdots r_{m-1} r_m} \bmod p$，然后，用户 U_m 利用在上一轮中从用户 U_i($1\leqslant i\leqslant n-1$)处得到的消息 $(g^{r_1 r_2 \cdots r_{m-2} r_{m-1}})^{f_i}$ 以及自己选取的随机数 r_m，计算 $k^{f_i} \bmod p$，最后用户 U_m 将 $k^{f_1}, k^{f_2}, \cdots, k^{f_{m-1}}$ 广播给协议所有的参与者。

(6) 用户 U_i($1\leqslant i\leqslant n-1$)在收到广播消息后，利用消息中的 k^{f_i} 和自己拥有的 r_i，计算共享的群组密钥 $k=(k^{f_i})^{r_i}$。

在效率方面，GDH. 3 群组密钥协商协议需要 $n+1$ 轮，每个用户需要发送 2 条消息，需要做 3 次幂运算(用户 U_m 需要 m 次)，此外，协议还需要 2 次广播。

为了减少用户之间的信息交换次数，Becker 和 Wille 提出了 2^d 立方体群组密钥协商协议[68]。

协议 10.48　2^d 立方体群组密钥协商协议。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q，g 是 Z_p 上的一个 q 阶乘法生成元。

假定参与协议的用户有 $n=2^d$ 个，这 2^d 个用户分别用 d 维向量空间 $\mathbf{GF}(2)^d$ 中的向量标识身份。同时，选定 d 维向量空间 $\mathbf{GF}(2)^d$ 中的一组基 $b_1, b_2, \cdots, b_d$。

具体步骤：

(1) 在第 1 轮中，对于每一个用户，如果他的标识向量为 v，那么，U_v 随机选取一个元素 r_v，$1\leqslant r_v\leqslant q-1$，计算 $t_{v,1}=g^{r_v} \bmod p$，并将 $t_{v,1}$ 发送给用户 U_{v+b_1}。

(2) 在第 1 轮消息交换完毕之后，用户 U_v 收到了用户 U_{v+b_1} 发送来的消息 $t_{v+b_1,1}$，用户 U_v 利用消息 $t_{v+b_1,1}$ 与自己选取的随机数 r_v，可以计算 $k_{v,1}=(t_{v+b_1,1})^{r_v} \bmod p$，这样，用户 U_v

和用户 U_{v+b_1} 有了临时共享密钥 $k_{v,1}$。

(3) 在第 j 轮($2\leqslant j\leqslant n-2$)中,用户 U_v 有一个临时共享密钥 $k_{v,j-1}$,他计算 $t_{v,j}=g^{k_{v,j-1}} \bmod p$,并将 $t_{v,j}$ 发送给用户 U_{v+b_j}。

(4) 在第 j 轮($2\leqslant j\leqslant n-2$)消息交换完毕之后,用户 U_v 收到了用户 U_{v+b_j} 发送来的消息 $t_{v+b_j,j}$,用户 U_v 利用消息 $t_{v+b_j,j}$ 与自己拥有的临时共享密钥 $k_{v,j-1}$,可以计算新的临时密钥 $k_{v,j}=(t_{v+b_j,j})^{k_{v,j-1}} \bmod p$,这样,用户 U_v 和用户 U_{v+b_j} 有了临时共享密钥 $k_{v,j}$。

(5) 在第 d 轮消息交换完毕之后,所有用户 U_v 可以计算新的临时密钥 $k_{v,d}$,这时,所有用户计算的临时密钥的值都相等,这样所有用户拥有相同的会话密钥 $k=k_{v,d}$。

在不用广播消息的条件下,通过 2^d 立方体群组密钥协商协议,2^d 个用户只需要 d 次消息交换,就可以获得群组会话密钥。

Burmester 和 Desmedt 提出了一个群组密钥协商协议[69],这个协议的设计比较精巧,在使用广播消息的条件下,发送的消息较少,计算量也比较小。

协议 10.49 Burmester-Desmedt 群组密钥协商协议。

参数设置:

系统选择并公布一个适合的大素数 p,并且使得 $p-1$ 的因子中有一个足够大的素数 q,g 是 Z_p 上的一个 q 阶乘法生成元。

具体步骤:

(1) 在第 1 轮中,每个用户 U_i($1\leqslant i\leqslant n$)随机选取一个元素 r_i,$1\leqslant r_i\leqslant q-1$,计算 $t_i=g^{r_i} \bmod p$,并将 t_i 发送给用户 U_{i-1} 和用户 U_{i+1}。

(2) 在第 2 轮中,用户 U_i($1\leqslant i\leqslant n$)收到了用户 U_{i-1} 发送来的消息 t_{i-1} 和用户 U_{i+1} 发送来的消息 t_{i+1},利用 t_{i-1} 和 t_{i+1},以及自己选择的随机数 r_i,用户 U_i 计算 $X_i=(t_{i+1}/t_{i-1})^{r_i}$,并将 X_i 广播给协议所有的参与者。

(3) 在收到所有的广播消息后,用户 U_i 可以计算共享的群组密钥 $k=(t_{i-1})^{mr_i}(X_i)^{m-1}(X_{i+1})^{m-2}\cdots X_{i-1}$。

执行完协议 10.49,所有用户的共享密钥为 $k=g^{r_1r_2+r_2r_3+\cdots+r_nr_1}$。

对所有的 i(所有的下标都是模 m 意义上的)定义

$$Y_i = t_i^{r_{i+1}} = g^{r_ir_{i+1}}$$

那么,对于所有的 i,有

$$X_i = \left(\frac{t_{i+1}}{t_{i-1}}\right)^{r_i} = \left(\frac{g^{r_{i+1}}}{g^{r_{i-1}}}\right)^{r_i} = \frac{g^{r_{i+1}r_i}}{g^{r_{i-1}r_i}} = \frac{Y_i}{Y_{i-1}}$$

下面的等式可以保证密钥计算正确进行。

$$\begin{aligned} t_{i-1}^{r_im}X_i^{m-1}X_{i+1}^{m-2}\cdots X_{i-2}^1 &= Y_{i-1}^m\left(\frac{Y_i}{Y_{i-1}}\right)^{m-1}\left(\frac{Y_{i+1}}{Y_i}\right)^{m-2}\cdots\left(\frac{Y_{i-2}}{Y_{i-3}}\right)^1 \\ &= Y_{i-1}Y_i\cdots Y_{i-2} \\ &= g^{r_{i-1}r_i+r_ir_{i+1}+\cdots+r_{i-2}r_{i-1}} \\ &= k \end{aligned}$$

上面介绍的群组密钥协商协议都是基于 Diffie-Hellman 密钥协商协议的,此外,还可以用其他方法来实现群组密钥协商协议。例如,Boyd 提出了一个利用一般的加密签名技术和 Hash 函数构造的群组密钥协商协议[70]。

协议 10.50 Boyd 群组密钥协商协议。

参数设置：

系统选择并公布一个安全的 Hash 函数 H。

假定用户 U_1 拥有自己的签名私钥和对应的验证公钥，其他用户 $U_2, \cdots, U_n$ 拥有用户 U_1 的验证公钥。用户 $U_2, \cdots, U_n$ 拥有自己的加密公钥以及对应的解密私钥，用户 U_1 拥有用户 $U_2, \cdots, U_n$ 的加密公钥。

具体步骤：

(1) 在第 1 轮中，用户 U_1 随机选取一个元素 r_1，用自己的签名私钥签名消息 $U_1, U_2, \cdots, U_n, h(r_1)$，并将 $U_1, U_2, \cdots, U_n, \mathrm{Sig}(U_1, U_2, \cdots, U_n, h(r_1))$ 广播给协议所有的参与者。

(2) 在第 2 轮中，用户 U_1 使用用户 $U_i(2 \leqslant i \leqslant n)$ 的加密公钥加密自己选择的随机数 r_1，得到 $E_i(r_1)$，并将 $E_2(r_1), E_3(r_1), \cdots, E_n(r_1)$ 广播给协议所有的参与者。

(3) 在第 3 轮中，用户 $U_i(2 \leqslant i \leqslant n)$ 收到用户 U_1 广播的两条消息，首先对于第二条消息 $E_2(r_1), E_3(r_1), \cdots, E_n(r_1)$，用户 U_i 利用自己的解密私钥解密消息中的元素 $E_i(r_1)$，得到随机数 r_1，然后用户 U_i 计算 $h(r_1)$，并利用用户 U_1 的验证公钥验证第一条消息 $U_1, U_2, \cdots, U_n, \mathrm{Sig}(U_1, U_2, \cdots, U_n, h(r_1))$ 中的签名是否正确，如果签名通过验证，用户 U_i 随机选取一个元素 r_i，并将 r_i 广播给协议所有的参与者。

(4) 收到所有的广播消息后，用户 $U_i(2 \leqslant i \leqslant n)$ 可以计算共享的群组密钥 $k = \mathrm{MAC}_{r_1}(h(r_2) \oplus h(r_3) \oplus \cdots \oplus h(r_n))$。

Boyd 群组密钥协商协议的安全性基于 MAC 函数和选定的 Hash 函数。由于 r_1 是秘密发送给协议的参与者，因此，其他人无法计算共享密钥。密钥的新鲜性是由 Hash 函数保证的。

在讨论两个参与方的密钥协商协议时，讨论了基于身份的密钥协商协议，同样，在建立群组密钥时，也可以构造基于身份的群组密钥协商协议。基于传统数论问题，Koyama 和 Ohta 构造了 3 个群组密钥协商协议[71]，此外，还有几个基于传统数论问题的群组密钥协商协议，但是，这些基于身份的群组密钥协商协议比较难以应用。下面介绍两个基于双线性对的群组密钥协商协议。

协议 10.51 Yao-Wang-Jiang 基于身份的群组密钥协商协议[72]。

参数设置：

系统中的可信服务器 TA 选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加法群 G_1 的一个生成元 P，并且要求群 G_1 和 G_2 上的离散对数问题是困难的。系统再选取一个双线性映射 $e: G_1 \times G_1 \rightarrow G_2$ 和一个将二进制串映射到 G_1 的单向函数 h、一个安全的 Hash 函数 H。TA 再选取自己的私钥 s 并计算对应的公钥 $R = sP$。

任何用户在使用 Yao-Wang-Jiang 基于身份的群组密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。如果一个用户的身份信息为 ID，TA 计算这个用户的公钥为 $Q = h(\mathrm{ID})$，私钥为 sQ。

在执行密钥协商协议之前，用户 $U_i(1 \leqslant i \leqslant n)$ 获取的公钥为 $Q_i = h(U_i)$，对应的私钥为 $S_i = sQ_i$。记符号 U 为 $U_1, U_2, \cdots, U_n$。

具体步骤：

(1) 在第 1 轮中，对于任何一个用户 $U_i(1\leqslant i\leqslant n)$，他首先随机选取一个元素 r_i，$1\leqslant r_i\leqslant q-1$，然后用户 U_i 计算 $E_i=r_iP$，$F_i=H(U, e(E_i, R))S_i+r_iR$，最后用户 U_i 将 E_i、F_i 广播给协议所有的参与者。

(2) 在第 2 轮中，对于任何一个用户 $U_i(1\leqslant i\leqslant n)$，他收到了其他用户 $U_j(1\leqslant j\leqslant n, j\neq i)$ 发来的消息 E_j 和 F_j，首先，用户 U_i 验证收到的 E_j 和 F_j 是否为 1，并且验证等式 $e\left(\sum_{j\neq i}F_j, P\right)=e\left(\sum_{j\neq i}H(U, e(E_j, R))Q_j+E_j, R\right)$ 是否成立，如果通过验证，用户 U_i 构造以下形式的 T，$T=H(U_1, E_1, \cdots, U_n, E_n)$，然后用户 U_i 计算 $Y_i=r_iT$，$X_i=r_i(E_{i+1}-E_{i-1}+T)$，最后用户 U_i 将 X_i、Y_i 广播给协议所有的参与者。

(3) 在第 3 轮中，对于任何一个用户 $U_i(1\leqslant i\leqslant n)$，他收到了其他用户 $U_j(1\leqslant j\leqslant n, j\neq i)$ 发来的消息 X_j 和 Y_j，首先，用户 U_i 验证等式 $e\left(\sum_{j\neq i}Y_j, P\right)=e\left(\sum_{j\neq i}E_j, T\right)$ 是否成立，如果通过验证，用户 U_i 计算 $Z_i=e\left(nr_iE_{i-1}+\sum_{j=0}^{n}(n-1-j)(X_{i+j}-Y_{i+j}), R\right)$，然后，用户 U_i 计算 $C_i=H(i, U, E_1, \cdots, E_n, X_1, \cdots, X_n, Y_1, \cdots, Y_n, Z_i)$，最后用户 U_i 将 C_i 广播给协议所有的参与者。

(4) 对于任何一个用户 $U_i(1\leqslant i\leqslant n)$，他收到了其他用户 $U_j(1\leqslant j\leqslant n, j\neq i)$ 发来的消息 C_j，用户 U_i 验证等式 $C_j=H(j, U, E_1, \cdots, E_n, X_1, \cdots, X_n, Y_1, \cdots, Y_n, Z_i)$ 是否成立，如果通过验证，用户 U_i 计算共享的群组密钥 $k=H(U, E_1, \cdots, E_n, X_1, \cdots, X_n, Y_1, \cdots, Y_n, Z_i, C_1, \cdots, C_n)$。

Yao-Wang-Jiang 基于身份的群组密钥协商协议是一个 3 轮的群组密钥协商协议，文献[72]中给出了协议的安全性证明。

协议 10.52 Zhou-Susilo-Mu 基于身份的群组密钥协商协议[73]。

参数设置：

系统中的可信服务器 TA 选择并公布一个适合的大素数 q、一个 q 阶加法群 G_1、一个 q 阶乘法群 G_2，以及加法群 G_1 的一个生成元 P，并且要求群 G_1 和 G_2 上的离散对数问题是困难的。系统再选取一个双线性映射 $e: G_1\times G_1\rightarrow G_2$ 和一个将二进制串映射到 G_1 的单向函数 h、一个将 G_2 中的元素映射到 1b 二进制串的安全的 Hash 函数 H_1 和一个输出为 1b 二进制串的安全的 Hash 函数 H_2。TA 再选取自己的私钥 s 并计算对应的公钥 $R=sP$。

任何用户在使用 Zhou-Susilo-Mu 基于身份的群组密钥协商协议之前需要向 TA 注册，从而获得身份的私钥。如果一个用户的身份信息为 ID，TA 计算这个用户的公钥为 $Q=h(\text{ID})$，私钥为 sQ。

在执行密钥协商协议之前，用户 $U_i(1\leqslant i\leqslant n)$ 获取的公钥为 $Q_i=h(U_i)$，对应的私钥为 $S_i=sQ_i$。

具体步骤：

(1) 在第 1 轮中，对于任何一个用户 $U_i(1\leqslant i\leqslant n)$，他首先随机选取 G_2 中的一个元素 δ_i，以及两个随机数 r_i 和 k_i，然后用户 U_i 计算 $P_{i,j}=H_2(e(S_i, Q_j)\cdot\delta_i)\oplus r_i$，$1\leqslant j\leqslant n, j\neq i$，最

后用户 U_i 将 δ_i，$P_{i,1}$，…，$P_{i,i-1}$，$P_{i,i+1}$，…，$P_{i,n}$，$H_3(r_i)\oplus k_i$，L_i 广播给协议所有的参与者，这里 L_i 是一个包含着 $P_{i,j}$ 如何与接收者相联系的信息。

(2) 对于任何一个用户 $U_i(1\leqslant i\leqslant n)$，他收到了其他用户 $U_j(1\leqslant j\leqslant n, j\neq i)$ 发来的消息 $D_j=(R_j, P_{j,1}, \cdots, P_{j,j-1}, P_{j,j+1}, \cdots, P_{j,n}, V_j, L_j)$，用户 U_i 首先利用 L_j 中的信息找到适当的 $P_{j,i}$，然后用户 U_i 计算 $k_j'=H_3(H_2(e(Q_j, S_i)\cdot R_j)\oplus P_{j,i})\oplus V_j$，最后用户 U_i 计算共享的群组密钥 $k=k_1'\oplus\cdots\oplus k_{i-1}'\oplus k_i\oplus k_{i+1}'\oplus\cdots\oplus k_n'$。

上面利用密钥协商技术来建立群组密钥，也可以使用密钥分配技术来建立群组密钥。一个简单的方法[69]是一个用户 U_1 首先和其他用户 $U_i(2\leqslant i\leqslant n)$ 分别利用 Diffie-Hellman 密钥协商协议建立一个临时密钥 k_i，然后用户 U_1 选取一个会话密钥 k，最后用户 U_1 用 k_i 加密会话密钥 k 并发送给用户 U_i。下面介绍一个带有认证功能的群组密钥分配协议。

协议 10.53　Hirose-Yoshida 群组密钥分配协议[74]。

参数设置：

系统选择并公布一个适合的大素数 p，并且使得 $p-1$ 的因子中有一个足够大的素数 q，g 是 Z_p 上的一个 q 阶乘法生成元。

假定用户 U_1，U_2，…，U_n 各自拥有自己的签名私钥和对应的验证公钥，并且拥有其他用户的验证公钥。

具体步骤：

(1) 在第 1 轮中，每个用户 U_1 随机选取一个元素 r_1，$1\leqslant r_1\leqslant q-1$，计算 $t_1=g^{r_1} \bmod p$，并将 t_1 广播给协议所有的参与者。

(2) 在第 2 轮中，用户 $U_i(2\leqslant i\leqslant n)$ 收到了用户 U_1 发送来的消息 t_1 后，随机选取一个元素 r_i，$1\leqslant r_i\leqslant q-1$，计算 $t_i=g^{r_i} \bmod p$，并将 t_i 发送给用户 U_1。

(3) 在第 3 轮中，用户 U_1 收到了所有用户 $U_i(2\leqslant i\leqslant n)$ 发送来的消息 t_i 后，在集合 $\{g^0, g^1, \cdots, g^{q-1}\}$ 中随机选取一个元素作为会话密钥 k，计算 $c_i=k(t_i)^{r_1}$，$2\leqslant i\leqslant n$，并将 c_i 发送给用户 U_i。

(4) 在第 4 轮中，用户 $U_i(2\leqslant i\leqslant n)$ 收到了用户 U_1 发送来的消息 c_i 后，利用自己的签名私钥计算 $\mathrm{Sig}_i(t_1, c_i)$，并将 $\mathrm{Sig}_i(t_1, c_i)$ 发送给用户 U_1。

(5) 在第 5 轮中，用户 U_1 收到了所有用户 $U_i(2\leqslant i\leqslant n)$ 发送来的消息 $\mathrm{Sig}_i(t_1, c_i)$ 后，利用每一个用户 $U_i(2\leqslant i\leqslant n)$ 的验证公钥验证签名 $\mathrm{Sig}_i(t_1, c_i)$ 是否正确，如果通过验证，用户 U_1 利用自己的签名私钥计算 $\mathrm{Sig}_1(t_i, c_i)$，并将 $\mathrm{Sig}_1(t_i, c_i)$ 发送给用户 U_i。

(6) 用户 $U_i(2\leqslant i\leqslant n)$ 收到了用户 U_1 发送来的消息 $\mathrm{Sig}_1(t_i, c_i)$ 后，利用每一个用户 U_1 的验证公钥验证签名 $\mathrm{Sig}_1(t_i, c_i)$ 是否正确，如果通过验证，用户 U_i 可以计算共享的群组密钥 $k=c_i/(t_1)^{r_i}$。

当用户建立群组密钥时，如果参与者比较多，协议的效率会比较差。为了提高群组密钥建立的效率，文献[75]中提出了分层的群组密钥协商协议。

协议 10.54　分层群组密钥协商协议(以 2 层为例)。

参数设置：

假定 n 个用户 U_1，U_2，…，U_n 想要建立共享的群组密钥。

系统根据选择使用的群组密钥协商协议公布适当的系统参数。

具体步骤：

(1) 在第 1 轮中,n 个用户依据计算能力、所在位置等因素分成若干个组,每组中选举一个头节点,这些头节点组成上一层网络。

(2) 在第 2 轮中,首先,在每一个小组中,用户利用群组密钥协商协议建立一个组内临时会话密钥,然后小组的头节点以组内临时会话密钥为参数,建立一个群组密钥。

(3) 在第 3 轮中,每个小组的头节点将建立的群组密钥安全地发送给该小组的每一个成员。

10.6 小结

在密钥交换协议方面目前已形成了大量各种各样的标准。国际标准组织(ISO)在密钥交换协议方面主要有两个系列标准,分别是 ISO/IEC 11770 和 ISO 9594—8/ITU X. 509。国际标准组织推出的 ISO/IEC 11770 包括 4 个部分:框架(1996)、用对称技术的机制(1996)、用非对称技术的机制(1999)和基于弱密钥的机制(2006)。在第二部分中,标准讲述了使用对称技术建立会话密钥,包括有服务器和没有服务器两种情形。在第三部分中,标准讲述了使用非对称技术建立会话密钥,包括密钥分配和密钥协商两种方式。在第四部分中,标准讲述了基于弱密钥技术建立会话密钥。此外,在 ISO/IEC 15946—3 标准(2002)中,讲述了基于椭圆曲线技术的密钥分配协议和密钥协商协议。国际标准组织推出的 ISO 9594—8 标准(1995),主要讲述公开密钥基础设施技术,其中的"强认证"部分讲述了 3 个密钥协商协议。

国际电信联盟(ITU)在密钥交换协议方面推出了几个标准,即 ITU X. 509 标准、ITU X. 1035 标准和 ITU X. 1151 标准等。ITU X. 509 标准(1995)和国际标准组织的 ISO 9594—8 标准一致。在 ITU X. 1035 标准中,讲述了基于口令的认证密钥交换协议。在 ITU X. 1151 标准(2007)中,为基于口令的密钥交换协议提供指南。

Internet 工程任务组(IETF)是一个关注 Internet 体系结构演变和 Internet 平稳运作的研究机构,在 IETF 发布的 RFC 系列文档中,推荐了一些密钥交换协议。例如,RFC 2412(1998)讲述了 Oakley 密钥协商协议;RFC 2631(1999)讲述了 Diffie-Hellman 密钥协议;RFC 4120(2005)讲述了 Kerberos 协议;RFC 4306(2005)讲述了互联网密钥协商协议;RFC 4346(2006)讲述了传输层安全协议等。

电子和电子工程师研究院(IEEE)推出的 IEEE P1363—2000 标准(2000),主要讲述了公开密钥算法和密钥协商协议。美国国家标准协会(ANSI)推出的与密钥交换协议相关的标准有 X9. 42 标准(2001)和 X9. 63 标准(2002),在这两个标准中,讲述了利用公钥技术建立会话密钥。

文献[76]中的第 10 章和第 11 章专门介绍了密钥分配和密钥协商协议,这是一本很好的教科书,本章部分协议的分析采用了文献[76]中的表述方式。关于密钥交换协议的研究和综述可进一步参阅文献[77]~[83]。本书第 2 篇的第 3~6 章也穿插讨论了多个密钥交换协议,个别协议可能与本章有重复,但为了完整性还是按本章的分类方式和格式对这些协议做了介绍。本章在写作过程中得到了姚刚副教授的大力支持,他提供了大量的相关材料,作者在此表示衷心的感谢。

参 考 文 献

[1] ISO. Information Technology—Security Techniques—Key Management—Part 2: Mechanisms Using Symmetric Techniques ISO/IEC 11770—2. International Standard. 1996.

[2] Satyanarayanan M. Integrating security in a large distributed system. ACM Transactions on Computer Systems. Vol. 7, No. 3, 247～280, 1989.

[3] Menezes A J, Van Oorschot P C, Vanstone S A. Handbook of Applied Cryptography. CRC Press. 1996.

[4] Needham R, Schroeder M D. Using Encryption for Authentication in Large Network of Computers. Communications of the ACM. Vol. 21, No. 12, 993～999, 1978.

[5] Denning D E, Sacco G M. Timestamps in Key Distribution Protocols. Communications of the ACM. Vol. 24, No. 8, 533～536, 1981.

[6] http://en.winkipedia.org/wiki/Kerberos_(protocol).

[7] Bellare M, Rogaway P. ProvabIy secure session key distribution: the three party case. In 27th Annual ACM Symposium on Theory of Computing, pages 57～66. ACM Press, 1995.

[8] Lowe G. Breaking and Fixing the Needham-Schroeder Public Key Protocol using FDR. Tools and Algorithms for the Construction and Analysis of Systems. pp. 147～166, Spring-Verlag, 1996.

[9] ISO. Information Technology—Security Techniques—Key Management—Part 3: Mechanisms Using Asymmetric Techniques ISO/IEC 11770—3. International Standard. 1999.

[10] Shoup V. On Formal Models for Secure Key Exchange. IBM Research Report RZ 3121. 1999. Available at http://www.shoup.net/papers/skey.pdf, 1999.

[11] ITU/ISO. Information Technology—Open Systems Interconnection—The Directory—Part 8: Authentication Framework, ITU-T Rec. X.509—ISO/IEC 9594-8. International Standard. 1995.

[12] Blake-Wilson B, Menezes A. Entity Authentication and Authenticated Key Transport Protocols Employing Asymmetric Techniques. Security Protocols—5th International Workshop. LNCS 1361, 137～158, Spring-Verlag, 1998.

[13] Beller M J, Yacobi Y. Fully-Fledged Two-Way Public Key Authentication and Key Agreement for Low-cost Terminals. Electronic Letters. Vol. 29, No. 11, 999～1001, 1993.

[14] Boyd C, Mathuria A. Key Establishment Protocols for Secure Mobile Communications: A Critical Survey. Computer Communications. Vol. 23, No. 5/6, 575～587, 2000.

[15] Diffie W, Van Oorschot P C, Wiener M J. Authentication and authenticated key exchanges, Designs, Codes and Cryptography, 2(1992), 107～125.

[16] Blom R. An optimal class of symmetric key generation schemes, Advances in Cryptology—Eurocrypt'84, springer-verlag, 1985, 335～338.

[17] Leighton T, Micali S. Secret-Key Agreement without Public-Key Cryptography. Advances in Cryptology—Crypto'93. LNCS 773, 456～479, Spring-Verlag, 1993.

[18] Diffie W, Hellman M E. New Directions in Cryptography. IEEE Transaction on Information Theory. Vol. 22, No. 6, 644～654, 1976.

[19] Diffie W, Van Oorschot P C, Wiener M J. Authentication and Authenticated Key Exchanges. Designs, Codes, and Cryptography. Vol. 2, No. 2, 107～125, 1992.

[20] Matsumoto T, Takashima Y, Imai H. On Seeking Smart Public-Key-Distribution Systems.

Transactions of the IECE of Japan. Vol. 69, No. 2, 99～106, 1986.

[21] Menezes A J, Qu M, Vanstone S A. Some New Key Agreement Protocols Providing Implicit Authentication. 2nd Workshop on Selected Areas in Cryptography. 22～32, 1995.

[22] Lim C H, Lee P J. A Key Recovery Attack on Discrete Log-Based Schemes using a Prime Order Subgroup. Advances in Cryptology—Crypto'97. LNCS 1294, 249～263, Spring-Verlag, 1997.

[23] Just M, Vaudenay S. Authenticated Multi-Party Key Agreement. Advances in Cryptology—Asiacrypt'96. LNCS 1163, 36～49, Spring-Verlag, 1996.

[24] Goss K C. Cryptographic Method and Apparatus for Public Key Exchange with Authentication. US Patent 4,956,863. 1990.

[25] Girault M. Self-certified public key, Advances in Cryptology—Crypto'92, springer-verlag, 1991, 490～497.

[26] Menezes A J, Qu M, Vanstone S A. Some New Key Agreement Protocols Providing Implicit Authentication. 2nd Workshop on Selected Areas in Cryptography. 22～32, 1995.

[27] Law L, Menezes A J, Qu M. An Efficient Protocol for Authenticated Key Agreement. Designs, Codes and Cryptography. Vol. 28, No. 2, 119～134, 2003.

[28] IEEE. P1363 Standard Specifications for Public-Key Cryptography. IEEE standard 1363 ～ 2000. 2000.

[29] National Security Agency. SKIPJACK and KEA Algorithm Specification. http://csrc. nist. gov/groups/ST/toolkit/documents/skipjack/skipjack. pdf, 1998.

[30] Maughan D, Schertler M, Schneider M, Turner J. Internet Security Association and Key Management Protocol (ISAKMP). The Internet Society, RFC 2408, 1998.

[31] Orman H. The OAKLEY Key Determination Protocol. The Internet Society, RFC 2412, 1998.

[32] Krawczyk H. SKEME: A Versatile Secure Key Exchange Mechanism for Internet. IEEE Symposium on Network and Distributed System Security. 114～127, IEEE Computer Society Press, 1996.

[33] Kaufman C. Internet Key Exchange (IKEv2) Protocol. The Internet Society, RFC 4306, 2005.

[34] Joux A. A One Round Protocol for Tripartite Diffie-Hellman. 4th International Symposium on Algorithmic Number Theory Symposium. LNCS 1838, 385～394, Spring-Verlag, 2000.

[35] Yao G, Feng D G. Pairwise Key Agreement Protocols Based on the Weil Pairing. Journal of Software. Vol. 17, No. 4, 907～914, 2006.

[36] Okamoto E. Key Distribution Systems based on Identification Information. Advances in Cryptology—Crypto'87. LNCS 293, 194～202, Spring-Verlag, 1987.

[37] Günther C G. An Identity-Based Key Exchange Protocol. Advances in Cryptology—Eurocrypt'89. LNCS 434, 29～37, Spring-Verlag, 1989.

[38] Smart N P. An Identity-Based Authenticated Key Agreement Protocol Based on the Weil Pairing. Electronic Letters. Vol. 38, No. 13, 630～632, 2002.

[39] Skim K. Efficient ID-Based Authenticated Key Agreement Protocol Based on the Weil Pairing. Electronic Letters. Vol. 39, No. 8, 653～654, 2003.

[40] Chen L, Kudla C. Identity Based Authenticated Key Agreement Protocols from Pairings. 16th IEEE Computer Security Foundations Workshop. 219～233, IEEE Computer Society, 2003.

[41] McCullagh N, Barreto P S L M. A New Two-Party Identity-Based Authenticated Key Agreement. The Cryptographers' Track at the RSA Conference 2005. LNCS 3376, 262～274, Springer-Verlag, 2005.

[42] Xie G. Cryptanalysis of Noel McCullagh and Paulo S. L. M. Barreto's Two-party Identity-Based Key Agreement. http://eprint.iacr.org/2004/308, 2004.

[43] Choo K K R. Revisit of McCullagh-Barreto Two-Party ID-Based Authenticated Key Agreement Protocols. http://eprint.iacr.org/204/343, 2004.

[44] Wang H J, Yao G, Jiang Q S. An Identity-Based Group Key Agreement Protocol from Pairing. 3rd International Conference on Availability, Reliability and Security. 532～537, IEEE Computer Society, 2008.

[45] Zhang F, Liu S, Kim K. ID-based One Round Authenticated Tripartite Key Agreement Protocol with Pairings. http://eprint.iacr.org/2002/122, 2002.

[46] Lomas T M A, Gong L, Saltzer J H, Needham R. Reducing Risks from Poorly Choosen Keys. ACM Operating Systems Review. Vol. 23, No. 5, 14～18, 1989.

[47] Bellovin S M, Merritt M. Encrypted Key Exchange: Password-Based Protocols Secure against Dictionary Attacks. IEEE Symposium on Research in Security and Privacy. 72～84, IEEE Computer Society, 1992.

[48] Bellare M, Pointcheval D, Rogaway P. Authenticated Key Exchange Secure against Dictionary Attacks. Advances in Cryptology—Eurocrypt 2000. LNCS 1807, 139～155, Spring-Verlag, 2000.

[49] Boyko C, MacKenzie P, Patel S. Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman. Advances in Cryptology—Eurocrypt 2000. LNCS 1807, 156～171, Spring-Verlag, 2000.

[50] MacKenzie P. More Efficient Password-Authenticated Key Exchange. The Cryptographers' Track at the RSA Conference 2001. LNCS 2020, pp. 361～377, Springer-Verlag, 2001.

[51] Jablon D J. Strong Password-only Authenticated Key Exchange. ACM Computer Communication Review. Vol. 25, No. 5, 5～26, 1996.

[52] Wu T. The Secure Remote Password Protocol. Network and Distributed System Security Symposium. http://www.isoc.org/isoc/conferences/ndss/98/wu.pdf, 1998.

[53] Kwon T. Ultimate Solution to Authentication via Memorable Password. IEEE P1363 Standards Group Contibution. 2000.

[54] Lucks S. Open Key Exchange: How to Defeat Dictionary Attacks without Encrypting Public Keys. Security Protocols—5th International Workshop. LNCS 1361, 79～90, Spring-Verlag, 1998.

[55] MacKenzie P, Patel S, Swaminathan R. Password-Authenticated Key Exchange Based on RSA. Advances in Cryptology—Asiacrypt 2000. LNCS 1976, 599～613, Spring-Verlag, 2000.

[56] Lee H, Sohn K, Yang H, Won D. The Efficient 3-Pass Password-Based Key Exchange Protocol with Low Computational Cost for Client. 2nd International Conference on Information Security and Cryptology. LNCS 1787, 147～155, Springer-Verlag, 2000.

[57] Gong L, Lomas M A, Needham R, Saltzer J H. Protecting Poorly Chosen Secrets from Guessing Attacks. IEEE Journal on Selected Areas in Communications. Vol. 11, No. 5, 648～656, 1993.

[58] Tsudik G, Van Herreweghen E. Some Remark on Protecting Weak Keys and Poorly-Chosen Secrets from Guessing Attacks. 12th Symposium on Reliable Distribution Systems. 136～142, IEEE Computer Society, 1993.

[59] Gong L. Optimal Authentication Protocols Resistant to Password Guessing Attacks. 8th IEEE Computer Security Foundations Workshop. 24～49, IEEE Computer Society, 1995.

[60] Byun J W, Jeong I R, Lee D H, Park C S. Password-Authenticated Key Exchange between Clients with Different Passwords. 4th International Conference on Information and Communications

Security. LNCS 2513, 134～146, Springer-Verlag, 2002.

[61] Chen L. A Weakness of the Password-Authenticated Key Agreement between Clients with Different Passwords Scheme. Circulated for consideration at the 27th SC27/WG2 meeting, 2003.

[62] Wang S, Wang J, Xu M. Weaknesses of a Password-Authenticated Key Exchange Protocol between Clients with Different Passwords. 2nd International Conference on Applied Cryptography and Network Security. LNCS 3089, 414～425, Springer-Verlag, 2004.

[63] Kim J, Kim S, Kwak J, Won D. Cryptanalysis and Improvement of Password-Authenticated Key Exchange Scheme between Clients with Different Passwords. International Conference on Computational Science and Its Applications. LNCS 3043, 895～902, Springer-Verlag, 2004.

[64] Yao G, Feng D G, Han X X. Improved Client-to-Client Password-Authenticated Key Exchange Protocol. 2nd International Conference on Availability, Reliability and Security. 564～571, IEEE Computer Society, 2008.

[65] Tang Q, Choo K-K R. Secure Password-Based Authenticated Group Key Agreement for Data-Sharing Peer-to-Peer Networks. 4th International Conference on Applied Cryptography and Network Security. LNCS 3989, 162～177, Springer-Verlag, 2006.

[66] Ingemarsson I, Tang D T, Wong C K. A Conference Key Distribution System. IEEE Transactions on Information Theory. Vol. 28, No. 5, 714～720, 1982.

[67] Steiner M, Tsudik G, Waidner M. Diffie-Hellman Key Distribution Extended to Group Communication. 3rd ACM Conference on Computer and Communication Security. 31～37, ACM Press, 1996.

[68] Becker K, Wille U. Communication Complexity of Group Key Distribution. 5th ACM Conference on Computer and Communication Security. 1～6, ACM Press, 1996.

[69] Burmester M, Desmedt Y. A Secure and Efficient Conference Key Distribution System. Advances in Cryptology—Eurocrypt'94. LNCS 950, 275～286, Spring-Verlag, 1995.

[70] Boyd C. On Key Agreement and Conference Key Agreement. 2nd Australasian Conference on Information Security and Privacy. LNCS 1270, 294～302, Springer-Verlag, 2006.

[71] Koyama K, Ohta K. Identity-Based Conference Key Distribution Systems. Advances in Cryptology—Crypto'87. LNCS 293, 175～184, Spring-Verlag, 1987.

[72] Yao G, Wang H J, Jiang Q S. An Authenticated 3-Round Identity-Based Group Key Agreement Protocol. 3rd International Conference on Availability, Reliability and Security. 538～543, IEEE Computer Society, 2008.

[73] Zhou L, Susilo W, Mu Y. Efficient ID-Based Authenticated Group Key Agreement from Bilinear Pairings. 2nd International Conference on Mobile Ad-hoc and Sensor Networks. LNCS 4325, 521～532, Springer-Verlag, 2006.

[74] Hirose S, Yoshida S. An Authenticated Diffie-Hellman Key Agreement Protocol Secure against Active Attacks. 1st International Workshop on Practice and Theory in Public Key Cryptography. LNCS 1431, 135～148, Springer-Verlag, 1998.

[75] Yao G, Ren K, Bao F, et al.. Making the Key Agreement Protocol in Mobile Ad Hoc Network More Efficient. 1st International Conference of Applied Cryptography and Network Security. LNCS 2846, 343～356, Springer-Verlag, 2003.

[76] Stinson D R. Cryptography—Theory and Practice(Third Edition),CRC Press.(密码学原理与实践. 冯登国 等译.北京:电子工业出版社,2009.)

[77] Van Tilburg J. Secret-key exchange with authentication, Computer Security and Industrial

Cryptography, State of the Art and Evolution, ESAT Course, May, 1991, springer-verlag, 1993, 71～86.

[78] Harn L, Yang S B. ID-based Cryptographic Schemes for user identification, digital signature and key distribution, IEEE J. Select. Areas Commun, Vol. 71, 757～760.

[79] Blundg C, D'Arco P. The key establishment problem. Lecture Notes in Computer Science, 2946 (2004), 44～90. (Foundations of Security Analysis and Design II.)

[80] Boyd C, Mathuria A. Protocols for Authentication and Key Establishment. Springer, 2003.

[81] NG E M. Security Models and Proofs for Key Establishment Protocols. Masters Thesis, University of Waterloo, 2005.

[82] Wang H B. Desired Features and Design Methodologies of Secure Authenticated Key Exchange Protocols in the Public-key Infrastructure Setting. Masters Thesis, University of Waterloo, 2004.

[83] Feng D G, Xu J. A New Client-to-Client Password-Authenticated Key Agreement Protocol. IWCC 2009: 63～76.

第 11 章　健忘传输协议

健忘传输(Oblivious Transfer)协议是一类部分秘密泄露协议，又称不经意传输协议，简称 OT 协议。OT 协议可用于实现安全多方计算，也是构建其他安全协议(如电话掷硬币协议)的基础模块之一。

11.1　基本 OT 协议

OT 协议是由 Rabin 于 1981 年首次提出的一种协议[1]，在 Rabin 提出的 OT 协议模型中，参见图 11.1，A 以 50% 的概率向 B 传送秘密。因此，B 有 50% 的机会收到秘密和 50% 的机会收不到秘密。另一方面，B 将知道他是否已收到秘密，而 A 则不知道他自己是否真的给 B 传送了秘密。这也是将这种协议称为健忘传输协议的理由。



图 11.1　Rabin OT 协议模型　　**图 11.2　二择一 OT 协议模型**

Even、Goldreich 和 Lempel 于 1985 年引入了另外一种可称为二择一(One-out-of-two)的 OT 协议[2]，简记为OT_1^2或$\binom{2}{1}-OT$。在此协议模型中，参见图 11.2，发送者拥有两个消息 m_0 与 m_1，接收者拥有输入 $e\in\{0,1\}$。协议运行结束后，接收者获得了消息 m_e 而不能得到关于 m_{1-e}的任何信息，而发送者没有获得关于 e 的任何信息。最基本的OT_1^2是 bit-OT_1^2，此时 $m_0,m_1\in\{0,1\}$。OT_1^2形式上与 Rabin OT 协议完全不同，但实际上它们是等价的[3]，即从其中一个出发可以构造另外一个。再者，由于 OT_1^2 形式简单且易于理解，因此从其被提出以后，人们对 OT_1^2 协议的研究大都针对OT_1^2及其扩展形式展开。

11.1.1　Rabin 的 OT 协议

Rabin 的 OT 协议要求接收者以 1/2 的概率收到发送者发送的秘密，而且只有接收者自己知道是否正确收到了发送者的秘密。Rabin 基于二次剩余给出了协议 11.1。

协议 11.1

(1) 发送者选择 Blum 整数 $n=pq$，$p\equiv q\equiv 3 \pmod 4$，并随机选择 $t\in Z_n^*$，将$(n,(-1)^s t^2)$发送给接收者。

(2) 接收者随机选择 $x\in Z_n^*$，计算并发送 $a=x^2 \bmod n$ 给发送者。

(3) 发送者随机在 a 的 4 个平方根(模 n)中选择一个，记为 b，然后将 b 发送给接收者。

(4) 接收者收到 b 后，验证 $b=\pm x$ 是否成立。若成立，接收者得不到任何关于 s 的信息；若不成立，接收者此时可以利用 $\gcd(x+b,n)$分解整数 n，从而通过检查得到$(-1)^s t^2$ 的二次剩余性。

因为 n 有两个不同的素因子，所以方程 $a=x^2 \bmod n$ 有 4 个根，这 4 个根可由中国剩余定理求得。由于在 a 的 4 个平方根(模 n)中，恰有两个同余于 $\pm x$，于是若发送者均匀选择 b，接收者将以 1/2 的概率恢复出秘密 s，这里需要假设接收方不具有直接计算二次剩余(模 n)的能力。另一方面，由于发送者不知道拥有哪个根，因此就不知道接收者是否收到了秘密 s。为便于区分，以下记 Rabin 的 OT 协议为 R-OT 协议。

11.1.2 二择一 OT 协议

Even、Goldreich 和 Lempel[2] 引入的二择一 OT 协议，其中发送者拥有两个比特消息 m_0 与 m_1，接收者拥有输入 $e\in\{0,1\}$。要求协议运行结束后，接收者获得了消息 m_e 而不能得到关于 m_{1-e} 的任何信息，而发送者没有获得关于 e 的任何信息。现在所说的 OT 协议若无特别声明，一般均指此类协议。Even 等人最初给出的 OT 协议需要公钥加密系统为基础，即假设协议双方共享一个公钥加密系统，因此所给协议比较简单，具体构造参见协议 11.2。

协议 11.2

假定发送者的输入为 $m_0, m_1\in M$；接收者的输入为 $e\in\{0,1\}$。

(1) 发送者随机选择密钥对 (K_E, K_D) 和 $m_0', m_1'\in M$；并将 (K_E, K_D) 和 $m_0', m_1'\in M$ 发送给接收者。

(2) 接收者随机选择 $m\in M$，并将 $x=E_{K_E}(m)\oplus m_e'$ 交给发送者。

(3) 发送者计算 $x_0=D_{K_D}(x\oplus m_0')$，$x_1=D_{K_D}(x\oplus m_1')$，然后发送 $m_0\oplus x_0$ 与 $m_1\oplus x_1$ 给接收者。

(4) 接收者计算 $m_e=x_e\oplus m'$。

由于 OT 协议是最基本的安全协议之一，因此在 Even 等人最初工作之后，吸引了众多研究者对此进行了大量的研究，提出各种形式的 OT 协议和许多具体的构造。

11.1.3 两种 OT 协议的等价性

尽管从形式上看，R-OT 协议与 OT_1^2 有很大的不同，但它们在一定意义下的确是等价的，也就是它们之间可以相互(以黑箱方式)归约。

1. R-OT $\Rightarrow OT_1^2$

假设有实现 R-OT 的协议 π，若允许双方仅以黑箱方式运行 π，则可实现 OT_1^2。具体参见协议 11.3。

协议 11.3

发送者的输入为 $m_0, m_1\in M$；接收者的输入为 $e\in\{0,1\}$，接收者的输出为 m_e。

(1) 发送者随机选择 $B=b_1\cdots b_{3n}\in\{0,1\}^{3n}$。

(2) 发送者与接收者独立运行 $3n$ 次 π，其中在第 j 次中发送者输入为 b_j，而接收者获得 $v_j=b_j$ 或 $v_j=\perp$(表示接收者未获得 b_j)。

(3) 设 $I_{yes}=\{j: v_j=b_j\}\subset\{1,\cdots,3n\}$。接收者随机选择两个子集 I_0 与 I_1，满足 $|I_0|=|I_1|=n$，而且 $I_e\subset I_{yes}$。最后将 (I_0, I_1) 发送给发送者。

(4) 发送者计算并发送 $x_0=m_0\oplus(\bigoplus_{i\in I_0} b_i)$ 和 $x_1=m_1\oplus(\bigoplus_{i\in I_1} b_i)$ 给接收者。

(5) 接收者计算 $m_e = x_e \oplus (\bigoplus_{i \in I_e} v_i)$。

要保证接收者至少能得到其中之一，则必须有 $|I_{yes}| \geqslant n$。由于在 R-OT 中，接收者只以 50%的概率获得正确的比特，根据 Chernoff 不等式可得 $\Pr[|I_{yes}| < n] < 2e^{-cn}$（这里 $c>0$ 为常数），也就是说接收者得不到任何一个的概率是可忽略的。另一方面，为保证接收者最多只能得到其中之一，则必须有 $|I_{yes}| < 2n$，同样根据 Chernoff 不等式，$\Pr[|I_{yes}| \geqslant 2n] < 2e^{-cn}$。因此，除了 m_e 之外，接收者获得 m_{1-e} 的概率也是可忽略的。

由于发送者不能确定在每次运行 R-OT 协议后，接收者是否得到了 b_j，因此在收到 (I_0, I_1) 后，发送者无法判定是 $I_0 \subset I_{yes}$ 还是 $I_1 \subset I_{yes}$，也就是说发送者不能区分接收者的选择。因此，接收者的安全性由 R-OT 协议保证。

2. $OT_1^2 \Rightarrow$ R-OT

假设存在实现 OT_1^2 的协议 π，由此可实现 R-OT 协议，具体参见协议 11.4。

协议 11.4

发送者拥有 $b \in \{0,1\}$，欲以 50%的概率发送给接收者。

(1) 发送者随机选择 $\sigma, r \in \{0,1\}$，并置 $m_\sigma = b, m_{1-\sigma} = r$。

(2) 接收者随机选择 $e \in \{0,1\}$。

(3) 发送者与接收者运行 OT_1^2，其中发送者的输入为 (m_0, m_1)，接收者的输入为 e。结果接收者获得 m_e。

(4) 发送者发送 σ 给接收者。若 $\sigma = e$，接收者得到 b；若 $\sigma \neq e$，接收者仅得到随机比特 r。

显然，只要发送者诚实地执行协议，则必有 $\Pr[\sigma = e] = \frac{1}{2}$。另外，$OT_1^2$ 协议 π 保证接收者没有获得 m_{1-e} 的任何信息，因此接收者必仅以 50%概率得到或未得到 b。

11.2 一般 OT 协议

Brassard、Crépeau 和 Robert[4] 于 1986 年提出了 OT_1^2 的一种自然的推广，记为 OT_1^n，此时发送者拥有 n 个消息 $m_0, \cdots, m_{n-1}$，接收者的选择为 $e \in \{0, \cdots, n-1\}$。协议完成后，接收者也仅仅得到了 m_e，而发送者并未获悉接收者所做的选择。OT_1^n 的实现一般有两种方法，一种是利用基本的 OT_1^2 进行构造，另一种是直接利用密码学方法来实现。更一般地，可以考虑 OT_k^n，此时接收者有 k 个选择，$0 \leqslant e_0, \cdots, e_{k-1} \leqslant n-1$，协议的运行结果是接收者从发送者的 n 个消息中得到自己选择的 k 个消息。Bellare 与 Micali[5] 首先研究了 OT_{n-1}^n 的情形，其后 Naor 与 Pinkas 等人对一般 OT_k^n 的实现进行了研究[6,7]。显然，OT_k^n 可以由 k 次独立运行 OT_1^n 来实现，但这样做效率往往不高。因此，如何实现高效的 OT_k^n 协议一直是人们关注的研究内容，而研究如何由基本 OT_1^2 协议（或小规模的 OT 协议）构造一般的 $OT_k^n (k \geqslant 1)$ 协议更是一个有意义的问题。

在 OT 协议中，发送者被认为愿意与接收者合作（否则，发送者可以发送虚假的信息，协议无法排除这种可能性），而一个不诚实的发送者除此之外还希望能了解接收者的选择。OT 协议作为一个两方协议，它的安全性可以采用模拟的思想进行定义，也即要求对真实环境中的任意一个敌手 A，存在理想模型中（存在可信第三方）的敌手 A'（称为模拟器），使得

现实环境下协议的运行(存在敌手 A)与理想模型下协议的运行(存在敌手 A')是不可区分的。满足此定义的协议称为完全可模拟的安全 OT 协议。其实,在 Beaver[8] 最早给出 OT 协议安全性的定义时,正是采用了这种思想。但为简单起见,Naor 对 OT 协议的安全性定义采用了一种相对简单的定义,即将发送者的安全性与接收者的安全性分割考虑,因此可称为半可模拟(Half-simulatable)的安全性(大多数 OT 协议的安全性采用此定义)。具体来讲,OT_1^2 协议应该满足以下 3 个条件。

(1) 正确性。若双方都正确地执行协议,协议运行的结果是接收者获得了 m_e。

(2) 发送者的安全性。对消息(m_0,m_1)的任意分布(假设(m_0,m_1)依照一定的分布来自于消息空间 $M_0\times M_1$)以及任意的概率多项式时间(PPT)敌手 A(接收者),存在理想模型中的 PPT 敌手(模拟器)A',使其输出与敌手 A 的输出是计算不可区分的。

(3) 接收者的安全性。对任意可能的发送者 A(敌手),以及任意的 $\sigma,\tau\in\{0,1\}$,A 不能区分接收者的选择是 σ 还是 τ,也就是在敌手 A 所收到的消息在接收者的两种任意的选择下是不可区分的。

对于更一般的 OT_k^n,其安全性也可类似地定义。与安全性定义有关的另外一个问题是敌手的模型,一种称为半诚实模型(Semi-honest Model);另一种称为恶意模型(Maci Model)。在半诚实模型中,所有的参与者(包括敌手)都正确地执行协议,敌手所能做的就是记录从交互过程以期获得更多的信息。在恶意模型中,敌手可以采取一切可能的行为。根据文献[9]或本书第 6 章可知,在半诚实模型下安全的协议,经过编译变换可以得到在恶意模型下安全的协议。因此,为简化问题,人们常常关注半诚实模型下的安全性。在以下的叙述中,若无特别声明,其安全性一般是指在半诚实模型下的安全性。

在一般的OT_k^n 中,接收者对 k 个消息的选择是一次完成的,而 Naor 与 Pinkas[10] 最早考虑 Adaptive-OT_k^n(为便于区分,简记为$OT_{k\times1}^n$),在这样的协议中,接收者可以动态地依次选择希望得到的消息,亦即接收者在获得 m_{e_i} 后才决定第 $i+1$ 次的选择 e_{i+1}。在$OT_{k\times1}^n$ 中,若仅要求满足半可模拟的安全性,则发送者可以实施称为 Selective-failure 的攻击。因此,在一些较复杂的环境下,半可模拟的安全性定义并不能满足人们对安全性的要求,由此引发人们对$OT_{k\times1}^n$的完全可模拟的安全性的关注。Camenisch 等人[11] 就不仅在 Oracle 模型下利用盲签名构造了一个有效完全可模拟的安全$OT_{k\times1}^n$,而且在标准模型中利用一个较强的复杂性假设同样也构造了安全可模拟意义下安全的$OT_{k\times1}^n$ 协议(他们的结果需要较强的非标准的 q-power DDH 和 q-strong Diffie-Hellman 假设)。Green 与 Hohenberger[12] 在相对较弱的复杂性假设下(称之为 DBDH 假设)实现了完全可模拟的一般的OT_k^n 和$OT_{k\times1}^n$。而 Lindell[13] 进一步在标准假设下(DDH 假设)实现了OT_1^2,其结果也可推广至OT_k^n。

由于协议运行环境的复杂性,而且 OT 协议作为最基本的安全协议之一,常常用于构造其他安全协议或安全多方计算协议,因此仅仅考虑协议独立运行时的安全性是不够的,需要研究 OT 协议在并发环境下的安全性和 UC 安全性。Garay[14] 最早研究了 OT 协议在并发环境下的安全性,给出并发环境下的安全性定义以及具体的实现方案。由于并发只考虑了 OT 协议自身的复合,它仍不能保证其在与其他协议进行任意复合时的安全性,因此仅有并发安全性有时也是不够的。最近,Peikert 等人[15] 在 CRS 模型中给出了一个实用的 UC 安全的 OT 协议的构造,并分析了在几个不同的密码学假设下的具体实现。此前,Canetti 等人[16,17] 也对 UC 安全的 OT 协议进行了研究。文献[18]在 CRS 模型中利用标准的复杂性

假设给出了具有 UC 安全性的 ECOT 协议。由于在一般模型(Plain Model)中不可能实现 UC 安全的(双方)OT 协议,Fischlin[19]在有多个参与方的环境下研究了在一般模式中如何构造 UC 安全的 OT 协议。

有关 OT 协议的另一个基本问题是 OT 协议存在所需假设问题。由于 OT 协议的完备性,即由 OT 协议可以实现对任何函数的安全多方计算,因此人们一直试图在更弱的假设下来构造 OT 协议。不幸的是,到目前为止人们并不知道其存在所需的最低假设。由文献[9]或本书第 6 章可知,利用零知识证明等工具,可以将半诚实模型中的安全协议转化为恶意模型中的安全协议,因此,一般来说人们只需关心在半诚实模型中 OT 协议存在的条件。在加强陷门置换存在的假设下 OT 协议存在,而且一般认为一般陷门函数不能保证 OT 协议的存在,但陷门置换也许可以满足 OT 协议的需要。最近,Haitner[20]利用稠密陷门置换构造了 OT 协议,Peikert 与 Waters[21]使用一种新的陷门函数(Lossy trapdoor functions)给出了相应的构造。Impageliazzo 与 Rudich[22]证明了密钥协商 KA(Key Agreement)协议不能以黑箱方式归约到 OWF(One-Way Functions),而 OT 协议隐含 KA 协议(黑箱归约意义下)[23],于是可知,OT 协议也不可能以黑箱方式归约到 OWF。Harnik 与 Naor[24]最近的研究表明,在一定的假设下(假设有 SAT 问题实例存在证据可恢复的压缩算法),单向函数隐含着 OT 协议的存在。

11.2.1 String-OT_1^2

Brassard 等人[4]于 1986 年提出了 String-OT_1^2协议的概念,它与OT_1^2唯一的不同是发送者的消息 m_0 与 m_1 均是 k 长比特串。设 $m_i=m_{i,1}\cdots m_{i,k}$,$i=0,1$。实现 String-$OT_1^2$的一个简单办法是运行 k 次安全的OT_1^2,其中在第 j 次的OT_1^2中,发送者输入$(m_{0,j},m_{1,j})$,而接收者的输入是 e。此方法虽然简单,但有缺陷:不诚实的接收者可以分别得到 m_0 与 m_1 的部分比特。Brassard 等人利用了一个很简单的方法,便克服了上述的缺陷,从而通过多次运行安全的OT_1^2构造了安全的 String-OT_1^2。具体协议参见协议 11.5。

协议 11.5

发送者的输入为 $m_0,m_1\in M$;接收者的输入为 $e\in\{0,1\}$,接收者的输出为 m_e。

(1) 发送者随机选择 $s_0=s_0^1\cdots s_0^n\in\{0,1\}^n$,$s_1=s_1^1\cdots s_1^n\in\{0,1\}^n$。

(2) 对任意的 $1\leqslant j\leqslant n$,发送者与接收者运行OT_1^2,这里发送者的输入为(s_0^j,s_1^j),接收者的输入为 e。接收者最终得到一个 n 长的串 z。

(3) 发送者随机选择两个 $k\times n$ 的比特矩阵 M_0 与 M_1,计算 $x_0=m_0\oplus M_0\cdot s_0$ 与 $x_1=m_1\oplus M_1\cdot s_1$,最后将$(M_0,x_0)$与$(M_1,x_1)$发送给接收者。

(4) 接收者计算 $m_e=x_e\oplus M_e\cdot z$。

显然,若接收者诚实,则在运行 n 次OT_1^2后,它必得到 s_e,从而可计算 m_e。同时,由于接收者不可能同时得到 s_0 和 s_1,因此也就不可能同时获得 m_0 和 m_1。若一个不诚实的接收者试图通过在OT_1^2协议中分别选择 s_0 与 s_1 的部分比特以获取更多的信息,它将得不到 m_0 和 m_1 中的任何一个。接收者的安全性由OT_1^2的安全性得到保证。

11.2.2 OT_k^n协议

Brassard 于 1986 年首先开始研究一般的 OT_1^n 协议($n\geqslant 2$),此时,发送者拥有 n 个输

入，$m_1,\cdots,m_n\in\{0,1\}^t$，其中 $m_j=x_{j,1}\cdots x_{j,t}$，接收者的私有输入为 e。协议结束后，接收者获得 m_e。在更一般的 OT_k^n 协议中，接收者拥有 k 个选择 $e_1,\cdots,e_k$，协议结束后，接收者获得 $m_{e_1},\cdots,m_{e_k}$，其安全性要求与OT_1^2类似。OT_k^n协议有两种模式，一种称为静态模式：接收者一次决定它的 k 个选择；另一种称为动态模式（为区别，一般记为$OT_{k\times1}^n$）：接收者一次选择一个，亦即接收者对 e_j 的选择依赖于 $m_{e_1},\cdots,m_{e_{j-1}}$。Naor 与 Pinkas 首先在 1999 年给出了一个静态的OT_k^n 协议，随后人们给出了许多OT_k^n 协议。下面首先介绍 Brassard[4] 给出的 OT_1^n 协议，他给出的协议基于二次剩余，具体协议参见协议 11.6。

协议 11.6

发送者拥有输入 $m_1,\cdots,m_n\in\{0,1\}^t$，其中 $m_j=x_{j,1},\cdots,x_{j,t}$，$1\leqslant j\leqslant n$；接收者的输入为 e，接收者的输出为 m_e。

（1）发送者随机选择两个不同的大素数 p 与 q，以及模 $N=pq$ 的非二次剩余 y。对任意的 $1\leqslant i\leqslant n$，$1\leqslant j\leqslant t$，随机选择 $r_{i,j}\in Z_N^*$，计算 $z_{i,j}=r_{i,j}^2y^{x_{i,j}}$。最后，发送者将 N、y 及 $z_{i,j}$ $(1\leqslant i\leqslant n,1\leqslant j\leqslant t)$发送给接收者。

（2）接收者随机选择$\{1,\cdots,n\}$上的置换 π，随机选择 $r'_{k,j}\in Z_N^*$，$b_{k,j}\in\{0,1\}$，令 $P_\pi=\{\tau_{k,j}=z_{i,j}r'^2_{k,j}y^{b_{k,j}}=r_{i,j}^2r'^2_{k,j}y^{x_{i,j}+b_{k,j}}:i=\pi^{-1}(k)\}_{\substack{1\leqslant k\leqslant n\\1\leqslant j\leqslant t}}$。接收者发送 P_π 和 $e'=\pi(e)$给发送者。

（3）发送者公开 $\tau_{e',j}(1\leqslant j\leqslant t)$的二次剩余特性。

（4）接收者计算 $m_e=x_{e,1}\cdots x_{e,t}$：若 $\tau_{e',j}$是一个二次剩余，则 $x_{e,j}=b_{e,j}$；否则，$x_{e,j}=1\oplus b_{e,j}$ $(1\leqslant j\leqslant t)$。

注意到 $z_{i,j}\in QR_N$ 当且仅当 $x_{i,j}=0$，依据二次剩余假设，接收者最多获得$m_1,\cdots,m_n$ 中的一个。因此，在半诚实模型中，发送者的安全性依赖于二次剩余假设，而接收者的安全性是无条件的。在一般的模型（恶意模型）中，情况要复杂一些。例如，恶意的接收者可以通过选择特殊的 P_π 以获得 $m_i\oplus m_j$，而恶意的发送者可以用一个二次剩余 y'代替 y 发送给接收者，这样 $\tau_{k,j}$与 $z_{i,j}$有相同的二次剩余特性，发送者据此可以获取接收者的选择 e。要使协议在恶意模型中安全，一个自然方法就是在协议中增加零知识证明，发送者证明 N、y 按协议的要求生成；接收者证明 P_π 由随机选择的$r'_{k,j}\in Z_N^*$，$b_{k,j}\in\{0,1\}$而得到，若如此将使协议变得很复杂。

下面介绍 Naor 与 Pinkas[6] 给出的静态OT_k^n协议，他们的实现方法是首先构造一个一般的OT_1^n，然后通过多次调用$OT_1^{\sqrt{n}}$协议实现OT_k^n。于是，协议的安全性归约到一般的$OT_1^{\sqrt{n}}$协议的安全性，而后者又可归约到基本OT_1^2协议的安全性。

在构造一般的OT_1^n 时，除了假设存在一个消息空间为$\{0,1\}^t$ 的OT_1^2协议之外，还需假设协议双方共享一个伪随机函数簇：$\{F_\alpha:\{0,1\}^m\rightarrow\{0,1\}^m\}_{\alpha\in\{0,1\}^t}$。另外，为方便起见，不妨假设 $n=2^l$，且对任意的 $1\leqslant i\leqslant n$，其对应的 l 位的二进制表示为$[i]_2=[i_1\cdots i_l]$。具体协议参见协议 11.7。

协议 11.7

发送者拥有输入 $m_1,\cdots,m_n\in\{0,1\}^t$；接收者的输入为 e，接收者的输出为 m_e。

（1）发送者随机选择$(\alpha_1^0,\alpha_1^1),\cdots,(\alpha_l^0,\alpha_l^1)$，其中 $\alpha_j^b\in\{0,1\}^t$，$b\in\{0,1\}$，$j\in\{1,\cdots,l\}$。接收者计算 $x_i=m_i\oplus(\oplus_{j=1}^lF_{\alpha_j^{i_j}}(i))$。

（2）发送者与接收者独立运行 l 次OT_1^2协议，其中在第 j 次中，发送者的输入为$(\alpha_j^0$，

α_j^1)，接收者的输入为 e_j。

(3) 发送者发送 $x_1,\cdots,x_n$ 给接收者。

(4) 接收者计算 $m_e=x_e\oplus(\bigoplus_{j=1}^{l}F_{\alpha_j^{e_j}}(e))$。

利用常用的复合模式(也称合成模式)，容易将接收者的安全性归约到OT_1^2协议中接收者的安全性。对于发送者的安全性，利用$\{F_\alpha:\{0,1\}^m\rightarrow\{0,1\}^m\}_{\alpha\in\{0,1\}^t}$的伪随机性，便容易地将其归约于$\mathrm{OT}_1^2$中的发送者的安全性。

有了以上的OT_1^t，可以由此构造一般的OT_k^n协议。具体构造过程参见协议 11.8。

协议 11.8

发送者拥有输入 $m_1,\cdots,m_n\in\{0,1\}^t$；接收者的输入为 $e_1,\cdots,e_k$，接收者的输出为 m_{e_1}，$\cdots,m_{e_k}$。

(1) 对 $j=1,\cdots,w$，发送者与接收者完成以下操作：

① 发送者随机选择密钥对$(R_1^j,C_1^j),\cdots,(R_{\sqrt{n}}^j,C_{\sqrt{n}}^j)$，以及$\{1,\cdots,n\}$上的置换 π_j。发送者将输入表示为$\{m_{\pi_{j,R}(I),\pi_{j,C}(I)}\}_{I=1}^n$，其中 $\pi_{j,R}(I)$表示 $\pi_j(I)$的前$\frac{l}{2}$b，$\pi_{j,C}(I)$表示 $\pi_j(I)$的后$\frac{l}{2}$b。发送者发送 π_j 给接收者。

② 发送者与接收者分别对$(R_1^j,\cdots,R_{\sqrt{n}}^j)$与$(C_1^j,\cdots,C_{\sqrt{n}}^j)$运行 k 次 $\mathrm{OT}_1^{\sqrt{n}}$协议。

(2) 发送者计算 $x_e=m_e\bigoplus_{j=1}^{w}(F_{R^j_{\pi_{j,R}(e)}}(e)\oplus F_{C^j_{\pi_{j,C}(e)}}(e))$，$e=1,\cdots,n$，并将 $x_1,\cdots,x_n$ 发送给接收者。

(3) 对 $e_1,\cdots,e_k$，接收者计算 $m_{e_v}=x_{e_v}\bigoplus_{j=1}^{w}\left(F_{R^j_{\pi_{j,R}(e_v)}}(e_v)\oplus F_{C^j_{\pi_{j,C}(e_v)}}(e_v)\right)$。

协议中参数 w 的选择应满足$\left(\frac{k^4}{\sqrt{n}}\right)^w<\delta$，此时协议将以 $1-\delta$ 的概率保证发送者的安全性。接收者的安全性可直接由OT_1^t的性质得到。

构造OT_k^n协议的另一个方法是直接从某个密码学假设出发进行构造，目前已给出许多种构造方法，但在此无法详尽所有的构造方案，只能选择介绍其中较为典型的构造方法。协议 11.9 是 Chu 等人[25]给出的协议，该协议需要 DDH 假设。

假设素数 p、q 满足条件 $p=2q+1$，$G_q\subset Z_p^*$ 为阶为 q 的循环子群，g、h 为其生成元。协议双方的公用参数为(g,h,G_q)，这里要求双方都不知道$\log_g h$。协议共有两轮，接收者首先利用一个承诺方案发送对选择的承诺，发送者收到此承诺后，以加密的方式发送所有的消息 $m_1,\cdots,m_n$，确保接收者只能解密与自己所承诺的选择对应的消息。具体协议参见协议 11.9。

协议 11.9

发送者拥有输入 $m_1,\cdots,m_n\in\{0,1\}^t$；接收者的输入为 $e_1,\cdots,e_k$，接收者的输出为 m_{e_1}，$\cdots,m_{e_k}$。

(1) 接收者随机选择 $a_i\in Z_q(i=0,1,\cdots,k-1)$，记

$$f(x)=a_0+a_1x+\cdots+a_{k-1}x^{k-1}+x^k$$
$$f'(x)=(x-e_1)\cdots(x-e_k)=b_0+b_1x+\cdots+b_{k-1}x^{k-1}+x^k$$

接收者发送 $A_i=g^{a_i}h^{b_i}(i=0,1,\cdots,k-1)$给发送者。

(2) 发送者随机选择 $r_j \in Z_q$，计算 $c_j=(g^{r_j}, m_j B_j^{r_j})(j=1,\cdots,n)$，其中 $B_j=A_0 A_1^j \cdots A_{k-1}^{j^{k-1}}(gh)^{j^k}=g^{f(j)}h^{f'(j)}$，并发送 $c_1,\cdots,c_n$ 给接收者。

(3) 接收者在收到 $c_j=(U_j,V_j)(j=1,\cdots,n)$后，计算 $m_{e_i}=V_{e_i}/U_{e_i}^{f(e_i)}(i=1,\cdots,k)$。

协议 11.9 的正确性是显然的。事实上，若双方均正确执行了协议，利用 $f'(e_i)=0$，显然就有

$$V_{e_i}/U_{e_i}^{f(e_i)} = m_{e_i} \cdot (g^{f(e_i)}h^{f'(e_i)})^{r_{e_i}}/g^{r_{e_i}f(e_i)} = m_{e_i} \quad i=1,\cdots,k$$

由于接收者所用的承诺方案具有完美的隐蔽性质，发送者从 $A_i(i=1,\cdots,n)$不能获得关于 $e_1,\cdots,e_k$ 的任何信息，因此接收者的安全性是无条件的。同时，在 DDH 假设下，该协议对半诚实的接收者是安全的。

协议 11.9 是否可以抵御恶意的接收者尚不能确定，尽管目前还不知道接收者如何可以进行欺骗，但也不能排除恶意的接收者可以通过选择一些特殊的 A_i 而获得额外的信息。通过对上述协议进行适当修改，得到了一个发送者无条件安全的协议。具体参见协议 11.10。

协议 11.10

发送者拥有输入 $m_1,\cdots,m_n\in\{0,1\}^t$；接收者的输入为 $e_1,\cdots,e_k$，接收者的输出为 $m_{e_1},\cdots,m_{e_k}$。

(1) 接收者随机选择 G_q 的生成元 h 及 $y\in Z_q$，计算 $B=g^y$。同时随机选择 $a_i\in Z_q(i=0,1,\cdots,k-1)$，记

$$f(x) = a_0 + a_1 x + \cdots + a_{k-1}x^{k-1} + x^k$$
$$f'(x) = (x-e_1)\cdots(x-e_k) = b_0 + b_1 x + \cdots + b_{k-1}x^{k-1} + x^k$$

令 $A_i=g^{a_i}h^{b_i}, C_i=g^{a_i y}h^{b_i}(i=0,1,\cdots,k-1)$。最后，接收者发送 $B,(A_0,\cdots,A_{k-1})$和$(C_0,\cdots,C_{k-1})$给发送者。

(2) 发送者随机选择$(r_j,s_j)\in Z_q\times Z_q(j=1,\cdots,n)$，计算

$$c_j = (X_i^{r_i}g^{s_i}, W_i^{r_i}B^{s_i} \oplus m_j) \quad j=1,\cdots,n$$

其中 $X_j=A_0A_1^j\cdots A_{k-1}^{j^{k-1}}g^{j^k}, W_j=C_0C_1^j\cdots C_{k-1}^{j^{k-1}}(gh)^{j^k}$。发送者发送 $c_1,\cdots,c_n$ 给接收者。

(3) 接收者在收到 $c_j=(U_j,V_j)(j=1,\cdots,n)$后，计算 $m_{e_i}=U_{e_i}^y\oplus V_{e_i}(i=1,\cdots,k)$。

协议 11.10 的正确性容易验证。若协议正常运行至结束，有

$$\begin{aligned}U_{e_j}^y\oplus V_{e_j} &= (g^{f(e_j)r_{e_j}}g^{s_{e_j}})^y \oplus (g^{f(e_j)y}h^{f'(e_j)})^{r_{e_j}}(g^y)^{s_{e_j}} \oplus m_{e_j}\\ &= g^{f(e_j)yr_{e_j}+ys_{e_j}} \oplus (g^{f(e_j)y})^{r_{e_j}}(g^y)^{s_{e_j}} \oplus m_{e_j}\\ &= g^{f(e_j)yr_{e_j}+ys_{e_j}} \oplus g^{f(e_j)yr_{e_j}+ys_{e_j}} \oplus m_{e_j} = m_{e_j}\end{aligned}$$

与 Chu 等人给出的协议类似，接收者首先对自己的选择进行承诺，所不同的是这里相当于进行了两次承诺，因此承诺方案的隐蔽性只能是计算意义下的(在 DDH 假设下)。不难将接收者的安全性归约至 DDH 假设。

设 $I=\{i:1\leqslant i\leqslant n, f(i)\neq 0\}$，显然 $|I|\geqslant k$。对任意的 $i\in I$，令 $a_i'=f(i)r_i+s_i, \sigma_i=\log_g h^{f(i)}$，则有

$$W_i^{r_i}B^{s_i} = (g^{f(i)y}h^{f'(j)})^{r_i}g^{ys_i} = g^{(f(i)r_i+s_i)y}(h^{f'(j)})^{r_i} = g^{a_i'y+\sigma_i r_i}$$

由于 $\sigma_i\neq 0$，故若发送者均匀选取 r_i，则 $W_i^{r_i}B^{s_i}$ 也是均匀分布的。由此可以保证发送者的安全性是无条件的。

协议的交互次数是度量协议复杂性的指标之一。最近，Mu 等人[26]给出了一个新的非

交互的OT_k^n协议。

11.2.3 $OT_{k\times1}^n$协议

$OT_{k\times1}^n$协议最早是由 Naor 与 Pinkas[10]提出的，与一般OT_k^n的安全性定义类似，对其安全性要求也是在半可模拟意义下。文献[10]在 DDH 假设下给出了$OT_{k\times1}^n$的实现协议，所给协议分为两个阶段：初始化阶段和传输阶段。在初始化阶段，发送者首先对拥有的 n 个消息进行承诺；在传输阶段，接收者按照自己的选择 e_i 获得相应的密钥，它使得接收者可以对应承诺值。随后，Ogata 和 Kurosawa[27]、Chu 等人[25]分别给出了 RO 模型下的$OT_{k\times1}^n$协议。最近，Camenisch 等人[11]进一步分析了半可模拟$OT_{k\times1}^n$协议的固有缺陷，在完全可模拟的安全性定义下，分别在 RO 模型和标准模型下给出了$OT_{k\times1}^n$协议，Laur 与 Lipmaa[28]利用陷门承诺方案与完全可模拟的OT_1^n构造具有固定 k 的$OT_{k\times1}^n$协议。

11.3 一般复杂性假设下的 OT 协议的构造

为了方便起见，可以将 OT 协议定义为一个两方安全计算问题：发送者与接收者的初始输入分别为$(m_1,\cdots,m_n)$与 e，发送者没有获得输出，接收者获得 m_e，可形式地定义如下：

$$OT_1^n((m_1,\cdots,m_n),e)=(\lambda,m_e)$$

其中 λ 表示空输出。由于 OT 协议的完备性，即基于 OT 协议可以实现安全多方计算，因此 OT 协议的存在性自然而然就成为人们关注的基本问题之一。既然OT_1^n协议是一个特殊的两方安全计算问题，其安全性自然可以沿用更强的安全性定义，即通过定义现实模型与理想模型的模拟范式定义其安全性，可称为完全可模拟(Fully-simulatable)的 OT 协议。本小节 OT 协议的安全性就是在这样的安全性意义下定义的。

Goldreich 利用加强陷门置换簇(Enhanced Trapdoor Permutations)构造了一类 OT 协议，人们希望在陷门置换簇存在或更弱的条件下实现 OT 协议，虽然随后不多的研究取得了一些结果，但仍未获得基于陷门置换簇的 OT 协议。事实上，已有结果表明由陷门置换簇构造 OT 协议在黑箱方式下是不可能的。

Goldreich 基于加强陷门置换簇构造的 OT 协议在半诚实模型中是安全的，有关的协议及其证明已在 6.2.2 小节介绍过。但在恶意模型中，Goldreich 基于加强陷门置换簇构造的 OT 协议是不安全的，因为此时接收者可以对所有的 j 都取 $y_j=f_\alpha(x_j)$，从而可获得全部的 m_j。为了获得一般模型(恶意模型)中安全的 OT 协议，可采用文献[9]中的一般方法，通过使用零知识证明等工具将半诚实模型中的 OT 协议变换为恶意模型中的 OT 协议，可以预见的是这样得到的 OT 协议就会很复杂。

最近，Ishai 等人[29]给出了相对简单的恶意模型中的安全OT_1^2协议，它们所需的假设依然是加强陷门置换簇的存在，同时还需要非交互的承诺方案 Commit(·;·)。文献[29]中的基本思想是先构造允许单方恶意模型下的OT_1^2协议，然后由此得到恶意模型中安全的OT_1^2协议。文献[29]中的构造利用了 Wolf 与 Wullschleger[30]最近证明的 OT 协议的对称性，即从 A 到 B 的OT_1^2(A 为发送者，B 为接收者)可以由从 B 到 A 的OT_1^2(B 为发送者，A 为接收者)得到。

假设存在协议 Π 实现从 B 到 A 的OT_1^2，下面介绍一个利用协议 Π 实现的从 A 到 B 的

OT_1^2协议。

协议 11.11

A 拥有输入 $m_0, m_1 \in \{0,1\}$；B 的输入为 $e \in \{0,1\}$，B 的输出为 m_e。

(1) B 随机选择 $r \in \{0,1\}$，并令 $s_0 = r, s_1 = r \oplus e$。

(2) A 计算 $e' = m_0 \oplus m_1$。

(3) A 与 B 运行协议 Π，其中 B 的输入为 (s_0, s_1)，A 的输入为 e'。Π 结束后，A 得到输出 τ。

(4) A 发送 $\sigma = m_0 \oplus \tau$ 给 B。

(5) B 计算 $m_e = \sigma \oplus r$。

该协议将从 A 到 B 的OT_1^2协议的实现，安全地归约到实现从 B 到 A 的OT_1^2。

Ishai 等人的构造从基本OT_1^2协议 11.12（为方便起见，将此基本协议记为 Π）出发，它与 6.2.2 小节介绍的基于加强陷门置换簇构造的 OT 协议（即协议 6.7）的区别仅在于对 y_j $(j \neq e)$的选择不同。在基于加强陷门置换簇构造的 OT 协议中，y_{1-e}由接收者自己独立随机选择，而在协议 11.12 中 y_{1-e}由双方共同决定（双方共同决定获取 y_{1-e}的抽样算法中的随机数）。这就为限制接收者的不诚实行为提供了可能。

协议 11.12

发送者拥有输入 $m_0, m_1 \in \{0,1\}$；接收者的输入为 $e \in \{0,1\}$，接收者的输出为 m_e。

(1) 接收者随机选择 $s_0 \in \{0,1\}^n$，$\rho \in \{0,1\}^{\text{poly}(n)}$，并将 $c = \text{Commit}(s_0; \rho)$发送给发送者，这里的多项式 poly()由承诺方案所决定。

(2) 发送者随机选择陷门置换，即$(\alpha, t) \leftarrow I(1^n)$，随机选择 $s_1 \in \{0,1\}^n$，发送 α 与 s_1 给接收者。

(3) 接收者随机选择 $s \in \{0,1\}^n$ 并令 $x_e = S(\alpha, s)$，计算 $y_{1-e} = S(\alpha, s_0 \oplus s_1)$，$y_e = f_\alpha(x_e)$。将$(y_0, y_1)$发送给发送者。

(4) 发送者发送 $\tau_0 = m_0 \oplus b(f_\alpha^{-1}(y_0))$与 $\tau_1 = m_1 \oplus b(f_\alpha^{-1}(y_1))$给接收者。

(5) 收到(τ_0, τ_1)后，接收者计算 $m_e = \tau_e \oplus b(x_e)$。

在该协议中，发送者从(y_0, y_1)中不可能获取 e 的任何信息（因为当 f_α 确是一个置换时，y_0 与 y_1 有着同样的分布），因此发送者唯一可以获取 e 的渠道就是通过 $c = \text{Commit}(s_0; \rho)$。这样自然将接收者的安全性归约到承诺方案的隐蔽性。对诚实的接收者来说，由于 $y_{1-e} = S(\alpha, s_0 \oplus s_1)$由双方共同决定，因此它只能获得唯一的 m_e。但是，由于协议并不要求公开 s_0（若在(4)之前对发送者公开 s_0，则发送者由此便可以获得 e），所以欺骗的接收者可以通过选择 x_{1-e}而计算 $y_{1-e} = f_\alpha(x_{1-e})$，从而便可保证自己同时也获得 m_{1-e}。因此，该协议也只是半诚实模型中安全的OT_1^2协议。下面利用该协议构造一个可以抵御恶意接收者的OT_1^2协议。

协议 11.13

发送者拥有输入 $m_0, m_1 \in \{0,1\}$；接收者的输入为 $e \in \{0,1\}$，接收者的输出为 m_e。

(1) 接收者随机选择 $r_1, \cdots, r_{2n} \in \{0,1\}$。

(2) 发送者随机选择 $2n$ 个比特对：$m_i^0, m_i^1 \in \{0,1\}$ $(i = 1, \cdots, n)$。

(3) 发送者与接收者并行地运行 $2n$ 个 Π，其中在第 i 个协议中发送者的输入为(m_i^0, m_i^1)；接收者的输入为 r_i。设运行记录分别为 $t_1, \cdots, t_{2n}$。

(4) 发送者与接收者运行安全的联合投币协议,双方获得输出 $q=q_1\cdots q_n\in\{0,1\}^n$。令 $Q=\{2i-q_i\}_{i=1}^n$,则 $|Q|=n$。

(5) 接收者发送 $\{(r_i,s_0^i)\}_{i\in Q}$ 给发送者,其中 s_0^i 是接收者在第 i 个基本协议的(1)中随机选择的 $s_0^i\in\{0,1\}^n$,即当 $i\in Q$,接收者公开第 i 个基本协议 Π 中的承诺。

(6) 对所有的 $i\in Q$,发送者验证公开的承诺。若验证通过,执行下一步;否则,终止协议。

(7) 对 $i\in\bar{Q}=\{1,\cdots,2n\}-Q$,接收者计算 $e_i=e\oplus r_i$ 随机,并发送 $\{e_i\}_{i\in\bar{Q}}$ 给发送者。

(8) 发送者计算并发送 $\tau_0=m_0\oplus(\oplus_{j\in\bar{Q}}m_j^{e_j})$ 与 $\tau_1=m_1\oplus(\oplus_{j\in\bar{Q}}m_j^{1-e_j})$ 给接收者。

(9) 收到 (τ_0,τ_1) 后,接收者计算 $m_e=\tau_e\oplus(\oplus_{j\in\bar{Q}}m_j^{r_j})$。

显然,该协议与基本协议 Π 有相同的接收者的安全性,因此若协议 Π 对半诚实的发送者是安全的,则该协议对半诚实的发送者也是安全的。关于发送者的安全性,文献[29]证明了该协议对恶意的接收者也是安全的。直观上,双方运行的 $2n$ 个基本协议 Π 中,其中有 n 个用于检测接收者的行为,剩余的 n 个用于实现不经意传输。若在基本协议 Π 中,y_{1-e} 确是按协议的规定通过抽样算法 $S(\alpha,s_0\oplus s_1)$ 得到,则接收者一定不能获得 m_{1-e}。因此,若接收者成功地同时获得了 m_e 与 m_{1-e},则该接收者一定在与 Q 对应的基本协议中诚实地执行了协议(否则发送者将终止协议),而在剩余的基本协议(即与 $\bar{Q}$ 所对应的协议)中都采取了欺骗行为。进一步,Q 由一个安全的联合投币协议产生,接收者完全无法事先猜测和控制 Q。因此,接收者在生成 Q 之前执行的 $2n$ 个协议 Π 中,其中仅有 n 个诚实地执行了协议,而且恰好与 Q 一致的概率为 $\frac{1}{2^n}$。由此说明接收者欺骗成功的概率是可忽略的,即协议对恶意的接收者也是安全的。

为了得到在一般的恶意模型中安全的 OT_1^2,还需要有可以抵御恶意发送者的 OT_1^2。具体协议参见协议 11.14。

协议 11.14

发送者拥有输入 $m_0,m_1\in\{0,1\}$;接收者的输入为 $e\in\{0,1\}$,接收者的输出为 m_e。

(1) 发送者计算 $e'=m_0\oplus m_1$。

(2) 接收者随机选择 $\sigma\in\{0,1\}$,令 $m_0'=\sigma,m_1'=\sigma\oplus e$。

(3) 双方运行协议 11.13,其中发送者充当接收者,输入为 e';接收者为发送者,输入为 m_0',m_1'。设发送者得到的输出为 τ。

(4) 发送者将 $m_0\oplus\tau$ 发给接收者。

(5) 接收者计算 $m_e=\sigma\oplus\tau$。

依据 OT_1^2 的对称性,该协议对恶意的发送者可以保证接收者的安全性,即协议可以抵御恶意的发送者。

协议 11.13 实际上将对半诚实的接收者安全的协议 11.12 转化为一个即使对恶意接收者也安全的协议,因此,若用协议 11.14 替换协议 11.13 中的协议 Π,将会得到一个可以抵御恶意接收者的 OT 协议。又由于协议 11.14 本身就可以抵御恶意的发送者,因此如此得到的协议就是一个在恶意模型中安全的 OT_1^2 协议。

协议 11.12 以及由此得到的协议都需要加强陷门置换簇存在的假设,已有很多文献试图在更弱的假设下实现 OT 协议,有兴趣的读者可参阅文献[15]、[20]、[31]和[32]。

11.4　分布式 OT 协议

分布式健忘传输(Distributed Oblivious Transfer, DOT)是 OT 协议的一种变形, Gertner[33]最早在 1997 年提出此概念,它与 PIR 和 SPIR 有着密切的关系。Naor 等人于 2000 年又重新提出了一个与 Gertner 的概念略微不同的 DOT 协议的模型,给出了安全性要求,并利用多项式插值的方法给出了相应的实现协议。随后,Blundo[34]与 Nikov[35]分别给出了更形式化的模型描述,并构造了比 Naor 与 Pinkas 给出的协议更一般的方案。Naor 及 Nikov 等人的协议不需要复杂性假设,随后,Tzeng[36]在标准的复杂性假设(DDH 假设)下也给出了实现协议。

DOT 协议中除发送者和接收者外,还有一组与接收者进行交互的服务器,记为 $S_1,\cdots,S_p$。发送者首先将自己拥有的消息 $m_1,\cdots,m_n$ 在一组服务器中分享,而接收者通过与全部或部分服务器进行交互,根据自己的选择 e 获得消息 m_e。由于 Naor 与 Pinkas 最早给出的 DOT 协议采用了门限方案,因此也可称为门限 OT(Threshold Oblivious Transfer),简记为 (k,p)-OT_1^n。(k,p)-OT_1^n 协议分为两个阶段:初始化阶段与健忘传输阶段。

(1) 初始化阶段。进行分享的阶段,发送者依据输入 $m_0,\cdots,m_{n-1}$ 与每个服务器 S_i 进行交互,结果是 S_i 得到程序(或函数)P_i。

(2) 健忘传输阶段。接收者选择其中至少 k 个服务器并用输入 e 与之进行交互,交互结束后,接收者可以成功恢复出 m_e,这里要求服务器之间没有任何通信。

DOT 协议的安全性要求如下。

(1) 发送者的安全性。当接收者与少于 k 个服务器进行交互,则接收者不能恢复任何消息;当接收者与 k 个或多余 k 个服务器交互时,它最多只能恢复出 $m_0,\cdots,m_{n-1}$ 中的一个消息。

(2) 接收者的安全性。存在 $1\leqslant k'\leqslant k$,任意少于 k' 个服务器即使合作也不能获得关于接收者的选择 e 的任何信息。

(3) 抗同谋性。存在 $1\leqslant k''\leqslant k$,接收者与少于 k'' 个服务器合作不能帮助接收者获得更多的信息。

Naor 与 Pinkas 首先考虑了较简单的比特 (k,p)-DOT_1^2 协议,发送者拥有输入 $m_0,m_1\in\{0,1\}$;接收者的输入为 $e\in\{0,1\}$。协议的一般结构如下:

(1) 发送者随机选择二元多项式 $Q(x,y)$,使得 $m_0=Q(0,0)$, $m_1=Q(0,1)$。

(2) 发送者将一元多项式 $Q(i,y)$ 发送给服务器 $S_i(i=1,\cdots,p)$。

(3) 接收者随机选择多项式 $Z(x)$,使得 $e=Z(0)$。这里要求由 $Q(x,y)$ 与 $Z(x)$ 定义的多项式 $R(x)=Q(x,Z(x))$ 的次数为 $k-1$。

(4) 接收者用 $Z(i)$ 询问服务器 S_i,得到 $R(i)=Q(i,Z(i))$。

(5) 当得到至少 k 个 $R(i)$,接收者重构 $R(x)$ 而得到 $m_e=R(0)$。

在具体实现时,可以选择较简单的多项式 $Q(x,y)$,这样会简化协议。而在文献[7]中,方案之一是 $Q(x,y)$ 选择了稀疏多项式 $Q(x,y)=P_x(x)+P_y(y)=\sum_{j=1}^{k-1}a_jx^j+b_1y+b_0$,从而得到以下的协议 11.15。

协议 11.15

发送者拥有输入 $m_0, m_1 \in \{0,1\}$；接收者的输入为 $e \in \{0,1\}$。

(1) 初始化阶段

① 发送者随机选择二元多项式

$$Q(x,y) = P_x(x) + P_y(y) = \sum_{j=1}^{k-1} a_j x^j + b_1 y + b_0$$

使得 $m_0 = b_0, m_1 = b_1 + b_0$。

② 发送者将一元多项式 $Q(i,y) = b_1 y + \left(b_0 + \sum_{j=1}^{k-1} a_j i^j\right)$ 发送给服务器 $S_i (i=1,\cdots,p)$。

(2) 健忘传输阶段

① 接收者随机选择 $k-1$ 次多项式 $Z(x)$，使得 $e = Z(0)$。

② 接收者选择至少 k 个服务器 $S_{i_1},\cdots,S_{i_l}$，用 $Z(i_j)$ 询问服务器 S_{i_j}，得到 $R(i_j) = Q(i_j, Z(i_j)) (j=1,\cdots,l)$。

③ 若 $l \geqslant k$，接收者重构 $R(x) = Q(x, Z(x))$ 而得到 $m_e = R(0)$。

在该协议中，接收者最多只能得到关于 m_0 与 m_1 的线性组合。事实上，若 $l < k$，则接收者不能得到信息；若 $l \geqslant k$（不妨就设 $l = k$），则接收者可以得到以下方程：

$$\begin{bmatrix} i_1^{k-1} & i_1^{k-2} & \cdots & i_1 & 1 & y_{i_1} \\ i_2^{k-1} & i_2^{k-2} & \cdots & i_2 & 1 & y_{i_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ i_k^{k-1} & i_k^{k-2} & \cdots & i_k & 1 & y_{i_k} \end{bmatrix} \begin{bmatrix} a_{k-1} \\ \vdots \\ a_1 \\ b_0 \\ b_1 \end{bmatrix} = \begin{bmatrix} Q(i_1, y_{i_1}) \\ Q(i_2, y_{i_2}) \\ \vdots \\ Q(i_k, y_{i_k}) \end{bmatrix}$$

由于系数矩阵共 k 行，其前 k 列恰好是范德蒙德矩阵，因此不论接收者如何选择 $(y_{i_1},\cdots,y_{i_k})$，接收者最多只能得到 b_0 与 b_1 的线性组合，也就是 m_0 与 m_1。另外，一个显然的事实是，只要接收者与一个服务器合作，则它可以同时得到 m_0 与 m_1，也就是说，抗同谋性的门限为 $k''=1$。

文献[7]声称上述协议对门限 $k'=k$ 时满足接收者的安全性，事实上 $k-1$ 服务器合作有可能获得接收者的选择 e。假设接收者选择的多项式为 $Z(x) = e + a_1 x + \cdots + a_{k-1} x^{k-1}$，$k-1$ 个服务器，不妨设为 $S_1,\cdots,S_{k-1}$，在分别得到 $Z(1),\cdots,Z(k-1)$ 后合作，用插值方法得到多项式 $Z'(x) = a_0' + a_1' x + \cdots + a_{k-2}' x^{k-2}$，使其满足 $Z'(i) = Z(i) (i=1,\cdots,k-1)$，于是必有 $e \neq a_0'$，从而若有 $a_0' \in \{0,1\}$，则可得 $e = 1 - a_0'$。因此，欲保证接收者的安全性，应有 $k' \leqslant k-1$。

文献[34]考虑了 (k,p)-DOT$_1^n$ 协议，发送者拥有 n 个秘密消息，$m_0,\cdots,m_{n-1} \in \mathrm{GF}(q)$，接收者的选择为 $e \in \{0,\cdots,n-1\}$。其安全性与上述协议的安全性类似，具体参见协议 11.16。

协议 11.16

发送者拥有 $m_0,\cdots,m_{n-1} \in \mathrm{GF}(q)$（有限域）；接收者的输入为 $e \in \{0,1\}$。

(1) 初始化阶段

① 发送者随机选择 n 元多项式

$$Q(x,y_1,\cdots,y_{n-1}) = \sum_{j=1}^{k-1} a_j x^j + b_0 + b_1 y_1 + \cdots + b_{n-1} y_{n-1}$$

满足 $m_0=b_0, m_1=b_0+b_1, \cdots, m_{n-1}=b_0+b_{n-1}$。

② 发送者将多项式 $Q(i,y_1,\cdots,y_{n-1})$ 发送给服务器 $S_i(i=1,\cdots,p)$。

(2) 健忘传输阶段

① 接收者随机选择 $n-1$ 个 $k-1$ 次多项式 $Z_1(x),\cdots,Z_{n-1}(x)$，使得 $(Z_1(0),\cdots,Z_{n-1}(0))$ 的第 e 分量为 1，其余全部为 0。定义 $k-1$ 次多项式

$$R(x) = Q(x,Z_1(x),\cdots,Z_{n-1}(x))$$

② 接收者选择至少 k 个服务器 $S_{i_1},\cdots,S_{i_l}$，用 $(Z_1(i_j),\cdots,Z_{n-1}(i_j))$ 询问服务器 S_{i_j}，得到 $R(i_j)=Q(i_j,S(i_j))(j=1,\cdots,l)$。

③ 若 $l\geqslant k$，接收者重构 $R(x)=Q(x,S(x))$ 而得到 $m_e=R(0)$。

文献[35]同样利用多项式插值的方法，得到比文献[34]中协议更一般的 (k,p)-DOT_1^n 协议。具体协议参见协议 11.17。

协议 11.17

发送者拥有输入 $m_0,\cdots,m_{n-1}\in \mathrm{GF}(q)$；接收者的输入为 $e\in\{0,1\}$。

(1) 初始化阶段

① 发送者随机选择 $k-1$ 次多项式 $B_0(x)$ 和 $n-1$ 个 v 次多项式 $B_1(x),\cdots,B_{n-1}(x)$，满足

$$m_0 = B_0(0), m_1 = B_0(0)+B_1(0),\cdots,m_{n-1} = B_0(0)+B_{n-1}(0)$$

这里要求 $k-1\geqslant v+k'-1$。

② 发送者构造 n 元多项式

$$Q(x,y_1,\cdots,y_{n-1}) = B_0(x)+B_1(x)y_1+\cdots+B_{n-1}(x)y_{n-1}$$

并将多项式 $Q(i,y_1,\cdots,y_{n-1})$ 发送给服务器 $S_i(i=1,\cdots,p)$。

(2) 健忘传输阶段

① 接收者随机选择 $n-1$ 个 $k'-1$ 次多项式 $Z_1(x),\cdots,Z_{n-1}(x)$，使得 $(Z_1(0),\cdots,Z_{n-1}(0))$ 的第 e 分量为 1，其余全部为 0。定义 $k-1$ 次多项式

$$R(x) = Q(x,Z_1(x),\cdots,Z_{n-1}(x))$$

② 接收者选择至少 k 个服务器 $S_{i_1},\cdots,S_{i_l}$，用 $(Z_1(i_j),\cdots,Z_{n-1}(i_j))$ 询问服务器 S_{i_j}。

③ 服务器 S_{i_j} 发送 $R(i_j)=Q(i_j,S(i_j))$ 给接收者 $(j=1,\cdots,l)$。

④ 若 $l\geqslant k$，接收者重构 $R(x)=Q(x,S(x))$ 而得到 $m_e=R(0)$。

该协议的正确性容易得到证明。由于 $B_1(x),\cdots,B_{n-1}(x)$ 的次数为 v，因此接收者即使与 v 个服务器合作，也不能获得除 m_e 以外的更多信息。值得注意的是，尽管接收者将自己的选择隐藏在 $k'-1$ 次多项式中，但利用与分析 (k,p)-DOT_1^2 协议相同的方法，$k'-1$ 个服务器合作是有可能获得有关 e 的信息。

除了以上基于多项式插值的门限 (k,p)-DOT_1^n 协议外，文献[35]还给出一般存取结构模型中的 DOT 协议。而文献[36]给出了基于 DDH 假设的 (k,p)-DOT_1^n 协议。

11.5 小结

健忘传输协议可用于实现安全多方计算，也是构建其他安全协议的基础之一。因此其研究受到人们的广泛关注。这些研究问题主要包括如何实现高效的 OT_k^n 协议？如何由基本OT_1^2协议（或小规模的 OT 协议）构造一般的 OT_k^n（$k \geq 1$）协议？如何刻画 OT 协议在并发环境下的安全性和 UC 安全性？是否可能在更弱的假设下构造 OT 协议？虽然本章已回答了部分问题并介绍了一些典型的相关工作，但是这些研究问题仍是人们一直甚至长期关注的研究焦点。

本章 11.3 节对初学者来说，可能不太好理解。这里将 OT 协议归结为一个两方安全计算问题，因此，为了更好地理解这部分内容，在学习这部分内容时可穿插阅读本书第 6 章的相关内容，这样做对于一些安全模型的理解也是有好处的。我们在这一领域也取得了一些其他研究成果，感兴趣的读者可参阅文献[37]～[39]。本章在写作过程中得到了李红达教授的大力支持，他提供了大量的相关材料，作者在此表示衷心的感谢。

参 考 文 献

[1] Michael O Rabin. How to Exchange Secrets by Oblivious Transfer. Technical Report. TR-81 (Harvard University: Aiken Computation Laboratory, 1981).
[2] Even S, Goldreich O, Lempel A. A Randomized Protocol for Signing Contracts, Communications of the ACM 28, 1985(6): 637～647.
[3] Crepeau C. Equivalence between two avours of oblivious transfers. In Advances in Cryptology—EUROCRYPT'87, LNCS, 350～354. Springer-Verlag, 1988.
[4] Brassard G, Cr'epeau C, Robert J M. All-or-nothing disclosure of secrets. In CRYPTO'86, Volume 263 of LNCS, Springer-Verlag, 1987.
[5] Bellare M, Micali S. Non-interactive oblivious transfer and applications. In Advances in Cryptology—CRYPTO'89, volume 435 of LNCS, 547～557, Springer-Verlag, 1990.
[6] Naor M, Pinkas B. Oblivious transfer and polynomial evaluation. In Proceedings of the 31th Annual ACM Symposium on the Theory of Computing (STOC'99), 245～254. ACM, 1999.
[7] Naor M, Pinkas B. Distributed oblivious transfer. ASIACRYPT'00,2000: 205～219.
[8] Beaver D. How to break a "secure" oblivious transfer protocol. In Advance in Cryptologyc'92, Volume 658 of LNCS, 285～296. Springer-Verlag, 1992.
[9] Goldreich O. Foundations of cryptography. Volume II, Basic Applications.
[10] Naor M, Pinkas B. Oblivious transfer with adqptive queries. In CRYPTO'99, Volume 1666 of LNCS, 573～590, Springer-Verlag, 1999.
[11] Camenisch J, Neven G, Shelat A. Simulatable adaptive oblivious transfer. In Advance in cryptology-EUROCRYPT'07, Volume 4515 of LNCS, 573～590. Springer-Verlag, 2007.
[12] Green M, Hohenberger S. Blind identity-based encryption and simulatable oblivious transfer. Cryptology ePrint Archive, Report 2007/235. http://eprint.iacr.org.
[13] Lindell A. Efficient Fully-simulatable oblivious transfer. In CT-RSA'08, Volume 4964 of LNCS, 52～70. Springer-Verlag, 2008.

[14] Juan A. Garay P. Mackenzie. Concurrent oblivious transfer. In 41nd FOCS, pp. 314～324, 2000.
[15] Peikert C, Vaikuntanathan V, Waters B. A framework for efficient and composable oblivious transfer. In CRYPTO'08, volume 5157 of LNCS, 554～571. Springer, 2008.
[16] Canetti R, Lindell Y, Ostrovsky R, Sahai A. Universally composable two-party and multi-party secure computation. In STOC, 494～503, 2002.
[17] Canetti R, Rabin T. Universal composition with joint state. In CRYPTO, 265～281, 2003.
[18] Garay J, MacKenzie P, Yang K. Efficient and universally composable committed oblivious transfer and applications. In TCC'04, Volume 2951 of LNCS, 297～316. Springer-Verlag, 2004.
[19] Fischlin M. Universally Composable Oblivious Transfer in the Multi-party Setting. In the Cryptographers' track at the RSA conference 2006, Volume 3860 of LNCS, 332～349. Springer-Verlag, 2006.
[20] Haitner I. Implementing oblivious transfer using collection of dense trapdoor permutations. In TCC'04, Volume 2951 of LNCS, 394～409. Springer-Verlag, 2004.
[21] Peikert C, Waters B. Lossy trapdoor functions and their application. In Proceedings of the 40th annual ACM symposium on Theory of computing, 187～196, 2008.
[22] Impageliazzo R, Rudich S. Limits on the provable consequences of one-way permutations. In 21nd STOC, 44～61, 1989.
[23] Gertner Y, Kannan S, Malkin T, Reingold O and Viswanathan M. The relationship between public key encryption and oblivious transfer. In 41nd FOCS, 325～338, 2000.
[24] Harnik D, Naor M. On the compressibility of NP instances and cryptographic applications, In 47nd FOCS, 719～728, 2006.
[25] Chu C K, Tzeng W G. Efficient k-out-of-n oblivious transfer schemes with adaptive and no adaptive queries. In Proceeding of Public Key Cryptography(PKC'05), Volume 3386 of LNCS, 172～183, Springer-Verlag, 2005.
[26] Mu Y, Zhang J, et al. Robust non-interactive oblivious transfer. IEEE Communication Letters, 7(4), 2003, 153～155.
[27] Ogata W, Kurosawa K. Oblivious key search. Journal of complexity, 20(2-3), 2004, 356～371.
[28] Laur S, Lipmaa H. ON security of sublinear oblivious transfer. Cryptology ePrint Archive, 2006.
[29] Ishai Y, Kushilevitz E, Yehudadell and Petrank E. Black-Box constructions for secure computation. In STOC, 99～108, 2006.
[30] Wolf S, Wullschleger J. Oblivious Transfer is Symmetric. Cryptology ePrint Archive, Report 2004/336. http://eprint.iacr.org.
[31] Cheong K Y, Koshiba T. Reducing complexity assumptions for oblivious transfer. Cryptology ePrint Archive, Report 2008/140. http://eprint.iacr.org.
[32] Harnik D, Naor M. On the compressibility of NP instances and cryptographic applications, In 47nd FOCS, 719～728, 2006.
[33] Gertner Y, Malkin T. Efficient distributed 1 out of n oblivious transfer. MIT Lab for Computer Science Technical Report, MIT-LCS-TR-714, April 1997.
[34] Blundo C, D'arco P, De Santis A, Stinson D R. New results on unconditionally secure distributed oblivious transfer. SAC'02,2002.
[35] Nikov V, Nikova S, Preneel B, and Vandewalle J. On the unconditionally secure distributed oblivious transfer. http://www.esat.kuleuven.ac.be.
[36] Tzeng W G. Efficient oblivioud transfer schemes. LNCS, 2274,2002:159～171.

[37] Li H D, Ji D Y, Feng D G, Li B. Oblivious Polynomial Evaluation. J. Comput. Sci. Technol. 2004, 19(4): 550～554.

[38] Li H D, Yang X, Feng D G, Li B. Distributed Oblivious Function Evaluation and Its Applications. J. Comput. Sci. Technol. 2004, 19(6): 942～947.

[39] Yao G, Feng D G. Proxy Oblivious Transfer Protocol. ARES. 2006: 190～197.

第 12 章　公平交换协议

在电子交易(如电子投标和拍卖、电子合同签署、电子匿名交易)、电子选举和电子支付等应用场景中交易的双方,往往其利益目标不一致。因此,在设计这些场景中所应用的安全协议时应遵循公平性原则,即参与协议的任何一方在协议执行的任何阶段都不能处于一种相对有利的地位,应保证交易双方都不能通过损害对方利益而得到它不应得的利益。本章主要从公平性的角度介绍一些安全协议,称为公平交换协议。

12.1　承诺方案

承诺方案是一个常用的基本模块。一个承诺方案 Com 是一个两方协议,它包含两个阶段:承诺阶段和打开阶段。在承诺阶段,承诺者 C 和接收者 R 先执行一个(可能需要交互)协议为一个具体的承诺方案生成所需的参数(为简单起见,仍把这个具体的方案记作 Com)。给定被承诺的消息 m,C 挑选一个随机比特串 r,计算 $c=\mathrm{Com}(m,r)$并将 c 发送给 R,这样就完成了承诺阶段。在打开阶段,C 将被承诺的消息 m 和承诺时使用的随机数 r 一并发送给 R,R 通过验证 $c=\mathrm{Com}(m,r)$是否成立来判断 m 是否为 C 在承诺阶段承诺的消息。

也可以把执行一次承诺方案想象成一个物理过程:在承诺阶段,承诺者 C 把某条消息写在纸上,然后把它锁进一个铁箱里,最后把这个铁箱交给接收者 R;在打开阶段,承诺者拿出钥匙打开铁箱,取出纸条交给接收者 R。

一般把承诺阶段需要交互的次数(C 和 R 发送消息次数的总和)称为承诺方案的轮数,而 1 轮的承诺方案也称为非交互的承诺方案,即在承诺阶段不需要接收者发送消息。

承诺方案通常被要求满足以下两个安全性条件。

(1) 隐藏性(Hiding,也称隐蔽性)。对于任意多项式规模的电路 R^*,给定任意的两个等长的比特串 m_0 和 m_1,对于随机选取的 r_0 和 r_1,在承诺阶段结束时,$c_0=\mathrm{Com}(m_0,r_0)$与 $c_1=\mathrm{Com}(m_1,r_1)$计算不可区分。

(2) 绑定性(Binding,也称约束性)。对于任意多项式规模的电路 C^*,它能计算出 m_0、m_1、r_0 和 r_1,使得 $m_0\neq m_1$ 并且 $\mathrm{Com}(m_0,r_0)=\mathrm{Com}(m_1,r_1)$的概率是可忽略的。这意味着承诺阶段结束后,承诺者不能改变被承诺的消息。

满足上述条件的承诺方案称为计算隐藏且计算绑定的承诺方案。虽然能从任意的单向函数构造出这样的方案,但已知的基于单向函数的方案在承诺阶段需要交互(准确地说,承诺阶段由两轮消息构成)。如果假设单向置换存在,则存在非交互的计算隐藏且计算绑定的承诺方案。

根据对敌手计算能力假定的不同,还有以下几种承诺方案。

统计隐藏且计算绑定的承诺方案。这种方案的绑定性要求和上面的条件(2)一样,但它满足更强的隐藏性要求:对于有着无限计算能力的接收者,上述 $c_0=\mathrm{Com}(m_0,r_0)$与 $c_1=$

$\mathrm{Com}(m_1, r_1)$统计不可区分。在构造方面，目前能从任意的单向函数构造交互的统计隐藏和计算绑定的承诺方案。如果假定离散对数问题是困难的，则存在两轮的统计隐藏且计算绑定的承诺方案。

计算隐藏且统计绑定的承诺方案。这种方案的隐藏性要求和上面的条件(1)一样，但它满足更强的绑定性要求：即使承诺者有着无限的计算能力，它能把一个承诺成功地打开成两个消息的概率也是可忽略的。如果假定单向函数存在，则存在两轮的计算隐藏和统计绑定的承诺方案。构造非交互的这样的方案则需要假设单向置换存在(或假定更强的数论问题)。

类似地，也可以定义完美隐藏且计算绑定的承诺方案和计算隐藏且完美绑定的承诺方案。如无特别声明，承诺方案一般指的是计算隐藏且计算绑定的承诺方案，而统计(完美)隐藏的承诺方案和统计(完美)绑定的承诺方案则分别指的是统计(完美)隐藏且计算绑定的承诺方案和计算隐藏且统计(完美)绑定的承诺方案。根据定义，读者可以验证不存在统计隐藏且统计绑定的承诺方案这一事实。

陷门承诺方案是一类特殊而重要的承诺方案，正如陷门承诺方案的名字所暗示的那样，陷门承诺方案的公共参数中含有陷门信息。对于不使用这些陷门信息的诚实的双方，它就是一个普通的承诺方案。如果拥有这个陷门信息，一个承诺者就可以将一个承诺打开成他想要的任意值。

一个陷门承诺方案满足以下的属性。

(1) 存在一个生成方案的 PPT 算法 G，给定 1^n 作为输入，输出对(pk, τ)，这里 pk 为方案的参数，τ 为陷门信息。

(2) 在没有使用陷门信息 τ 的情况下，由 pk 指定的承诺方案 $\mathrm{Com}_{\mathrm{pk}}$是一个标准的完美(或统计)隐藏的承诺方案。

(3) 给定陷门信息 τ，则存在一个高效的 PPT 算法能将任意一个承诺打开成任意指定的值。

下面介绍两个典型的承诺方案。

12.1.1 比特承诺方案

一般地，比特承诺方案通过一个函数 $f: \{0,1\} \times X \to Y$ 来实现，这里 X 和 Y 是两个有限集。$b \in \{0,1\}$的密文随机地在集合$\{f(b,x): x \in X\}$中取值。比特承诺方案应能满足以下两个特性：

(1) 隐藏性：对 $b \in \{0,1\}$，接收者不能从 $f(b,x)$确定出 b 的值。

(2) 绑定性：发送者能“打开”$f(b,x)$，即发送者能通过揭露辅助值 x 使接收者相信 b 是唯一可能被加密的值。

如果发送者想承诺任何比特串 s，那么他可以通过分别独立地承诺 s 的每一个比特来完成。比特承诺方案可记为 $f(s,k)$。

目前已有很多比特承诺方案，实现方法也很多，这里仅介绍一个基于 Goldwasser-Micali 概率加密算法实现的比特承诺方案。

首先来介绍 Goldwasser-Micali 概率加密算法[1]。

设 $n=pq$，p 和 q 是素数。选择一个正整数 $t\notin QR(n)$ 且 $\left(\frac{t}{n}\right)=1$，$QR(n)$ 表示模 n 的二次剩余之集，$\left(\frac{t}{n}\right)$ 表示 t 关于模 n 的 Jacobi 符号。公开 n、t，保密 p 和 q。$P=C=Z_2^n$，$K=\left\{(n,t,p,q)\mid n=pq,p\text{ 和 }q\text{ 为素数},t\notin QR(n),\left(\frac{t}{n}\right)=1\right\}$。

对 $K=(n,t,p,q)$，定义加密变换为 $E_K(x,r)=y=(y_1,y_2,\cdots,y_n)$，其中 $y_i=t^{x_i}r_i^2 \bmod n$，$1\leqslant i\leqslant n$，$x=(x_1,x_2,\cdots,x_n)\in P$，$\boldsymbol{r}=(r_1,r_2,\cdots,r_n)$ 是随机选择的一个向量。

解密变换为：$D_K(y)=(x_1,x_2,\cdots,x_n)$，其中

$$x_i=\begin{cases}0, & y_i\in QR(n)\\ 1, & y_i\notin QR(n)\end{cases},\quad 1\leqslant i\leqslant n,y=(y_1,y_2,\cdots,y_n)\in C$$

知道 n 的分解的用户可通过下列方法确定 $y_i\in QR(n)$ 还是 $y_i\notin QR(n)$。

(1) 计算 $\left(\frac{y_i}{p}\right)=y_i^{(p-1)/2} \bmod p$，$\left(\frac{y_i}{q}\right)=y_i^{(q-1)/2} \bmod q$。

(2) $y_i\in QR(n)\Leftrightarrow\left(\frac{y_i}{p}\right)=1,\left(\frac{y_i}{q}\right)=1$。

接下来介绍一个基于 Goldwasser-Micali 概率加密算法的比特承诺方案。

设 $n=pq$，p 和 q 都是素数，$m\in\overline{QR}(n)=\{x\mid(x/p)=(x/q)=-1\}$，公开 n 和 m，n 的分解只有发送者知道。设 $X=Y=Z_n^*$，$f(b,x)=m^bx^2 \bmod n$。发送者通过选择一个随机数 x 加密 b，加密结果为 $y=f(b,x)$。当发送者想“打开”y 时，它揭露值 b 和 x，接收者验证 $y=m^bx^2 \bmod n$。

假定二次剩余问题是不可行的，那么 $f(b,x)$ 没有泄露关于 b 和 x 的任何信息。所以该方案满足隐藏性。

现在来说明该方案满足绑定性。若该方案不满足绑定性，则存在 $x_1,x_2\in Z_n^*$，使得 $mx_1^2\equiv x_2^2 \bmod n$，这样 $m\equiv(x_2x_1^{-1})^2 \bmod n$。说明 $m\in QR(n)$，即 m 是一个二次剩余，这与 $m\in\overline{QR}(n)$ 相矛盾。

12.1.2 Pedersen 承诺方案

对比特的承诺也可以推广到对消息的承诺，Pedersen 承诺方案[2]就是一个对消息进行承诺的方案。Pedersen 承诺方案是一个两轮的具有完美隐藏的承诺方案。

设 n 为安全参数，p 和 q 为两个大素数，使得 $p=2q+1$，$|q|=n$，G_q 为 Z_p^* 的一个阶为 q 的子群，g 是 G_q 的一个生成元。

Pedersen 承诺方案中的承诺过程如下：接收者首先发送一个 G_q 中的随机元素 h；承诺者收到 h 后，为了承诺一个值 y，他随机选取承诺密钥 rE_RZ_q，计算 $C=g^yh^r \bmod p$ 并把 C 发送给接收者。如果要打开承诺 C，承诺者只需把 y 和 r 发送给接收者即可。

Petersen 承诺方案的完美隐藏性不需要依赖任何困难性假设，但它的计算绑定性依赖于求离散对数的困难性，即离散对数假设。

Pedersen 承诺方案可被修改成一个陷门承诺方案。如果承诺者能得到陷门信息 x 使得 $h=g^x \bmod p$，那么它就能将一个承诺 $C=g^yh^r \bmod p$ 打开成任意的值 y'。在打开阶段，

它计算 $r'=\frac{y-y'+rx}{x} \bmod q$，发送 y'和 r'。

12.2 电子选举协议

在民主社会中，选举总是使用一个锁着的箱子，该箱子的顶部有一个可放入选票的小缝，每次选举的选票是由选举委员会特制的。在选举时，人们将填好的选票一张接一张地投入选票箱。这样做的目的是确保选票的匿名性。随着计算机技术和通信技术的发展，信息处理和分配的量和速度都迅猛增加，电子选举已成为可能。

从理论上来讲，一个选举协议是具有秘密输入值的多成员的计算，它使得输出的正确性是可检验的。从实际应用来讲，一个选举协议可通过加密、盲签名和别的密码技术来实现。

一个理想的电子选举协议应满足以下一些特性。

(1) 完全性：所有合法选票都能被正确地统计(计数)。

(2) 合理性：不诚实的选举者不能扰乱选举。

(3) 秘密性：所有的选票都是秘密的，即能保护个人隐私。

(4) 不可重复性：任何选举者都不能投两次票。

(5) 合法性：只有经许可的选举者才能投票。

(6) 公平性：任何事情都不能影响选举。

(7) 可验证性：所有选举者都能检验他们的选票在最后的表中是否被统计上。

目前人们已经提出了许多选举协议，在这些协议中，有的满足上述全部特性；有的满足上述部分特性；有的除满足上述全部或部分特性外，还满足别的一些特性。本节只介绍一个使用盲签名技术设计的满足上述全部特性的选举协议，称为 FOO 选举协议[3]。

在 FOO 选举协议中，主要有 3 种参加者，即选举者、一个选举管理中心和一个计票者(计票者可能由一个公开的广告牌来代替)。该协议的设计需要 3 种密码技术，即比特承诺技术、普通的数字签名技术和盲数字签名技术。

在 FOO 选举协议中，选举管理中心所使用的盲数字签名协议可以是任何一种盲数字签名协议，为叙述方便，这里选择基于 RSA 的盲数字签名协议。

预备阶段：选举者 V_i选择并填写一张选票 v_i，随机选择一个密钥 k_i，用比特承诺方案 f 加密 v_i，即计算 $x_i=f(v_i,k_i)$。随机选择一个盲因子 $r_i\in Z_p{}^*$ 盲化 x_i，即计算 $e_i=r_i^e H(x_i) \bmod n$ 并对 e_i 签名得 $s_i=\sigma_i(e_i)$。然后将(ID_i,e_i,s_i)发送给选举管理中心 A。其中 σ_i表示 V_i的签名方案，ID_i为 V_i的身份证，H 是一个公开的单向函数。

颁发选举证书阶段：选举管理中心 A 检查选举者 V_i有无权力选举，如果 V_i无权参加选举，则 A 拒绝给 V_i颁发选举证书；否则，A 检查 V_i是否已经申请过一个选举证书，如果 V_i已申请过，A 拒绝再给 V_i颁发；否则，A 检查 s_i是否是消息 e_i的合法签名，如果是则 A 对 e_i签名，即计算 $d_i=e_i{}^d \bmod n$ 并将 d_i发送给 V_i，作为 A 颁布给 V_i的选举证书。在颁发选举证书阶段末，A 宣布已获得选举证书的选举者的总数并公布包含有(ID_i,e_i,s_i)的一张表。

投票阶段：选举者 V_i通过对 d_i脱盲恢复 A 对 x_i的签名 $y_i=d_i/r_i=H(x_i)^d \bmod n$。$V_i$检查 y_i是否是 A 对 x_i的合法签名，如果不是，V_i向 A 证明(x_i,y_i)不合法，并另选一个 v_i来重新获取选举证书，否则 V_i匿名地将(x_i,y_i)发送给计票者 C。

收集选票阶段： 计票者 C 通过使用 A 的签名验证方程检查 y_i 是否是 x_i 的合法签名。如果是，C 将 (l, x_i, y_i) 填入表 12.1 中，其中 l 是 (x_i, y_i) 的编号。在所有的选举者投完票后，C 公开该表。

表 12.1　收集选票阶段的选票表

序号	选票和附加信息
1	x_j, y_j
⋮	⋮
l	x_i, y_i
⋮	⋮

公开阶段： 选举者 V_i 检查选票的数量是否等于选票者的数量。如果不相等，要求其选票未被收入的选举者公开他们加密时使用的 r_i。V_i 检查他的选票是否列在表中，如果 V_i 没有列在表中，V_i 公开 (x_i, y_i)，选举中心 A 检验 V_i 的选票是否合法。合法的选举者可以要求将他的选票列入选票表。表上有序号 l 的 V_i 将 (l, k_i) 匿名地发送给 C。

统计选票阶段： 计票者 C"打开"选票 x_i 的承诺，恢复选票 v_i 并检查 v_i 是否是合法的选票。C 统计(计数)选票并宣布选举结果(表 12.2)。

表 12.2　统计选票阶段的选票表

序号	选票和附加信息
1	x_j, y_j, k_j, v_j
⋮	⋮
l	x_i, y_i, k_i, v_i
⋮	⋮

12.3　智力扑克协议

智力扑克类似于普通扑克[4]，只是没有牌也没有口头通信，参加者之间的所有交换都必须用消息完成。任一牌手都有可能想欺骗对方。

公平比赛至少要满足以下要求。

(1) 比赛必须从"公平发牌"开始。假定牌手们通过交换一系列消息实现了发牌，则发牌结果应满足下述条件：

① 牌手应知道自己手中的牌，但不知道其他人的牌。

② 手中的牌应不相连贯。

③ 每位牌手手中牌的分布应是相同的。

(2) 比赛中，牌手可能要从剩下的牌中补抓几张牌，补抓牌的结果也应像(1)中所述的那样是公平的。牌手还必须能将牌出示给对方而不泄露他们手中其余牌的秘密。

(3) 比赛结束时，牌手们应能检验比赛是否公平，以及他们的对手有没有骗人。特别是要能检查赢家是否作弊。

为叙述方便起见，假定只有两个牌手 A 和 B 玩牌，每个人都有一个秘密密钥，这些密钥在比赛结束以后才公开。设 E_A 和 D_A 分别表示 A 的加密变换和解密变换。E_B 和 D_B 分别

表示 B 的加密变换和解密变换。A 和 B 的加密变换必须是可换的，即对于任何消息 M，$E_A(E_B(M))=E_B(E_A(M))$。

以消息 $M_i(1\leqslant i\leqslant 52)$表示 52 张牌。B 开始发牌，发牌协议如下。

(1) B 对 52 个消息 $M_i(1\leqslant i\leqslant 52)$加密，得到 52 个密文消息：$E_B(M_i)(1\leqslant i\leqslant 52)$，而后他随机地洗这一副加了密的牌并将其送给 A。

(2) A 随机地选出 5 个密文消息，并送回给 B，B 将它们解密后留在手中。

(3) A 再随机地选出 5 个密文消息 $C_i=E_B(M_{j_i})(1\leqslant i\leqslant 5)$，他用自己的密钥对这 5 个消息再加密，得到 $C_i'=E_A(C_i)(1\leqslant i\leqslant 5)$，而后，将这 5 个通过两次加密的消息发送给 B。

(4) B 用他的密钥对每个消息 $C_i'(1\leqslant i\leqslant 5)$进行解密，得到 $D_B(C_i')=D_B(E_A(E_B(M_{j_i})))=D_B(E_B(E_A(M_{j_i})))=E_A(M_{j_i})$。他将这 5 个消息送给 A，而后 A 用他的密钥将它们解密并留在手中。

在比赛过程中，可重复上述过程来添牌。

因为在模算术下指数是可换的，所以可使用 RSA 密码算法来实现智力扑克协议。但用 RSA 密码算法来实现智力扑克协议会有少量信息被泄露。这主要是因为指数变换 $f(x)=x^e \bmod n$ 将二次剩余仍变为二次剩余的缘故。这个特性可被用来标记某些牌，比如所有的“A”。一种解决方法是把所有的数都变成二次剩余，这是可能的，因为一个模素数 p 的非二次剩余乘以一个模素数 p 的非二次剩余就是一个模素数 p 的二次剩余。

12.4 公平掷币协议

Blum[4] 于 1982 年采用比特承诺协议给出了一种通过网络或电话实现 A 和 B 之间公平掷币(Coin Flipping)的方法。其协议如下。

(1) A 用任何一个比特承诺方案承诺一个 bit 发送给 B。

(2) B 试图对此 bit 进行猜测，并将其猜测值发送给 A。

(3) A 向 B 披露他的承诺 bit。如果 B 的猜测正确，则 B 为赢家；否则为输家。

显然为了做到公平，A 必须在 B 猜测前将他掷硬币的结果发送给 B，且在得到 B 的猜测后不能重新掷硬币。B 必须在做猜测前不能得知 A 掷硬币的结果。其关键问题是设计一种协议，B 可以叫“正面”或“反面”，A 可以按一种方式掷硬币，使每个人都有 50%的机会获胜。现在已有很多种实现方法，包括利用单向函数，采用公钥密码体制，采用平方根，利用模素数指数和 Blum 整数等均可实现掷硬币协议。这里介绍一个利用 Blum 整数实现掷硬币的协议。

Blum 掷硬币协议如下。

(1) A 选出两个大素数 p 和 q，使得 $n=pq$ 是一个 Blum 整数，并将 n 发送给 B。

(2) B 检验 n 是否为素数、素数幂或偶数，如果是，则 A 就是骗人并认输；否则，B 取一 x，并将 $a=x^2 \bmod n$ 送给 A。

(3) A 计算 a 的 4 个根，随机地取出一个送给 B。

(4) 若 B 可分解 n，他就获胜。

之所以选择 $n=pq$ 为 Blum 整数，即 $p \bmod 4=q \bmod 4=3$，是为了便于 A 求解 $a=x^2 \bmod n$ 的根。实际上 n 可以是任何两个大素数 p 和 q 之积。

12.5 比较数的大小协议

Salomaa 于 1990 年给出了一个可以比较两个整数的大小而又不泄露具体数值的协议[5]，而 Yao 的百万富翁问题就是此问题的一个特例[6]。

假定 $1\leqslant i,j\leqslant 100$，A 知道 i，B 知道 j，A 和 B 均采用公钥加密算法。比较 i 和 j 大小的协议执行过程如下。

(1) A 选择一个随机大整数 x，用 B 的公钥加密得 $c=E_B(x)$。

(2) A 计算 $c-i$，并将结果发送给 B。

(3) B 计算下面 100 个数：$y_u=D_B(c-i+u)$，$1\leqslant u\leqslant 100$。

其中，D_B 表示 B 的解密变换。B 选择大素数 p(p 应小于 x，但 B 不知道 x，A 要将 x 的量级告诉 B)，计算下面 100 个数：$z_u=y_u \bmod p$ $(1\leqslant u\leqslant 100)$，检验是否对所有的 $u\neq v$ 有 $|z_u-z_v|\geqslant 2$ 和对所有的 u 有 $0<z_u<p-1$；如果不是，则 B 重新选择另一个素数试算，直到满足上述条件为止。

(4) B 将下面的数列依次发送给 A：

$$z_1, z_2, \cdots, z_j, z_{j+1}+1, z_{j+2}+1, \cdots, z_{100}+1, p$$

(5) A 检验接收数列中的第 i 个数是否同余于 $x \bmod p$；如果是，则 A 知道 $i\leqslant j$；否则有 $i>j$。

(6) A 将结论告诉 B。

协议的第(3)步中 B 所做的工作保证在第(4)步的数列中的数没有重复；否则，若 $z_a=z_b$，则 A 可以推知，$a\leqslant j\leqslant b$。该协议的一个缺点是：A 先于 B 知道结果，若他不愿意将真实结果告诉 B，B 也无可奈何。

可以将上述协议推广到一组人，通过计算机网络按一定逻辑排序，逐个比较他们的秘密数的大小，也可以将上述协议用于比较两个人的年龄大小、讨价还价、谈判、仲裁和“爱好匹配”等。

12.6 公平认证密钥交换协议

秘密通信的主要目的是阻止非授权方获取合法通信方之间传递的秘密信息，其研究焦点是在各授权方之间建立安全信道以防范非授权第三方(窃听方或主动攻击方)获得秘密信息，在信息安全领域通常可以归结为认证密钥交换(AKE)协议问题。在 AKE 协议模型中，考虑的敌手是某非授权的第三方，而对授权方(合法通信的双方或多方)的行为并没有什么限制，即授权方可以自由揭示任何有关通信信息，没有对此采取任何技术防范手段。在政府、军事和外交等领域，这个问题显得并不特别突出。但当今社会已经步入网络时代，隐私权问题已经成为一个相当广泛而核心的概念；另一方面，目前像网上购物或通过登录特定网站获取服务代理商的服务这样的网上交易业务日益普遍，关于服务商是否提供过某特定服务这样的纠纷也逐渐增多。如何平衡保护个人隐私和确保网上交易公平之间的矛盾已经成为一个不容忽视的问题。为解决该问题，Feng 等人提出了“公平认证密钥交换(FAKE)”的思想[7]，系统规划了 FAKE 协议的形式化安全模型，并给出了可证明安全的具体设计实例。

本节主要介绍这一工作。

12.6.1 公平认证密钥交换的基本思想

在具体阐述 FAKE 的基本思想之前,无妨设想以下的应用场景:无论在何地,客户 C 因某种商务或个人需要,需要通过个人计算机终端登录某服务器 S 以获取相应服务。一般做法是:C 利用自己的 Smart 卡(公钥证书)或口令远程登录服务器,通过执行一个安全的 AKE 协议,使得 C 和 S 之间共享一个秘密会话密钥 K,然后在 K 的保护下 C 从 S 获得具体服务(如加密通信)。

显然,如果 AKE 协议是安全的,那么除了 C 和 S 之外的任何第三方不足以对通信内容的机密性、认证性构成威胁。但问题是:作为信任方之一的服务器 S 往往具有向第三方提供"数字证明"的能力:证实某时某刻客户 C 曾经向 S 申请了某具体服务,这很可能严重威胁了客户 C 的隐私权。产生这个问题的原因显然在于 S 可以不受任何限制地泄露有关客户 C 的信息。

文献[8]中提出使用"可否认的密钥交换协议"来解决这个问题,基本想法是把文献[9]中提出的"可否认的认证"扩展为"可否认的密钥交换协议",即设计密钥交换协议时,既要考虑满足一般意义下的 AKE 协议的安全性,同时还要满足可否认性:消息的发送方或所有者在必要时可以否认曾经发送过该消息(另一方不具有向第三方证实曾经发生过会话的能力),但文献[8]中并未提出这类具体协议的实例,只是证明了基于公钥密码的 Internet 密钥交换协议 SKEME[10] 满足可否认性。并特别指出,当前多数 AKE 协议并不满足可否认性,因为为了抵抗诸如密钥替换之类的攻击,往往需要在协议会话中加入用户身份信息,而且广泛使用数字签名、消息认证码等技术,这都使得协议会话记录与用户身份绑定。

可否认的密钥交换协议的重要应用意义还在于:合法通信双方执行协议的结果是达成共享会话密钥 K,然后在进一步会话中使用 K 通过对称密码算法加密传递或认证消息;而如果会话是可否认的,那么进一步使用 K 保护的会话显然也是可否认的(因为会话密钥 K 和用户身份 ID 不再绑定)。因此,秘密通信的可否认性可以归结为 AKE 协议的可否认性。

尽管文献[8]提出的可否认的 AKE 协议的思想具有重要的使用价值,但还存在一些缺陷,主要是没有考虑平衡通信公平性和保护个人隐私之间的矛盾。无妨仍然继续考虑前面提到的例子:在客户 C 和服务器 S 会话结束之后,有可能一段时间之后 C 与服务商(服务器 S)产生有关 S 是否曾经提供网络服务的纠纷,这时 C 需要能够向第三方(如法官)证明某时某刻的确在 C 和 S 之间发生过上述秘密会话,则称为协议的公平性,并把这样的 AKE 协议称为"公平认证密钥交换(FAKE)协议"。

就 FAKE 协议而言,只有客户 C 能够提出协议会话发生的"证明",因此客户具有绝对意义下的"否认"权利,服务商只具有"部分否认"权利,这主要是出于侧重保护客户隐私的需要。公平交换问题一直是信息安全研究领域中的一个重要的基础问题,也有着非常现实的应用需求,如在各方互不信任的环境中进行数据交换。而我们提出的公平认证密钥交换思想与一般的公平交换问题的不同之处在于:着重强调保护个人隐私(可否认性)和客户权利(必要时举证),这在网上交易这类应用中有重要的应用价值。从前面的例子可以看出,文献[8]提出的"可否认的 AKE 协议"在保护客户权利方面有明显缺陷:发生纠纷时任何一方也无法提供数据交换通信的证明。最为"公平"的认证密钥交换协议是:利用一个可信或半可

信的(semi-trusted)第三方 T(裁决方)来处理客户 C 和服务商 S 之间的纠纷。这种技术路线原则上是可行的,但引入第三方 T 的做法很可能会破坏 C 的隐私,毕竟 T 不同于一般的密钥证书认证机构,适当可信的 T 并不容易得到,也增加了实现代价。另外,在实际应用中,完全意义上的公平交换往往并不必要[11]。

至此,可以提出关于 FAKE 协议的基本应用轮廓:立足实际应用,不采用任何可信第三方(当然公钥证书认证机构往往还是需要的);客户 C 发起和服务器(服务商)S 的 FAKE 协议,然后 C 利用作为协议结果的会话密钥 K 从 S 那里获得进一步的服务。这里除了要确保 FAKE 满足一般意义上的密钥交换协议的安全性[12]之外,还要确保:C 具有完全意义上的可否认性,S 满足部分可否认性,即在 C 不揭示秘密证据的条件下,C 和 S 各自均可以伪造(在第三方看来)合法的会话;必要时 C 可以向第三方提供协议会话曾经发生的数字证明,一旦该证明得以揭示,通信双方的身份均与协议记录得以绑定,因此合法通信双方均不再具有伪造协议会话的能力(公平性),当然就更不能伪造或否认基于会话密钥的进一步的秘密通信。

应该指出,FAKE 协议中的公平性是“相对”的,发起方(客户 C)由于控制了证据,因此事实上具有额外的权力:C 可以自主决定何时、向谁释放证据。这与求助于第三方的公平协议是有区别的。但在现实世界中存在很多技术或非技术方法确保 C 不会滥用这种权力;就实际应用而言,往往纠纷是关于某时某刻客户是否从服务商那里获得了某具体服务,作为弱势方的 C 为了保护自己的合法权利,当然会出示“证据”。因此 FAKE 协议是解决平衡保护客户隐私和确保各方合法权利之间矛盾的一种可行的技术解决方案。

12.6.2 FAKE 协议的安全模型

FAKE 协议是在互不信任环境中执行的特殊类型的 AKE 协议,其安全性内涵主要包含 3 层含义:作为一般 AKE 协议的安全性,即 mBJM-AK 安全性(主要是会话密钥的机密性和可认证性);可否认性;公平性。下面分别阐述这些概念。

1. mBJM 模型和模块化证明技术

有关 AKE 协议形式化安全模型的相关研究结果有很多,这方面最为系统的研究结果可参见文献[12]~[14],基本安全目标是确保会话密钥的机密性和可认证性。最早由文献[12]提出的 AKE 协议形式化安全模型是一种现实模型,即协议只定义在现实世界中,而安全性是通过会话密钥与随机数的计算不可区分性来定义的,定义了敌手拥有的 5 种 Oracle 来形式化敌手可能发动的各种攻击。文献[13]则进一步建议在设计复杂协议时采用简单、富有吸引力的模块化设计原则,一般称为 BCK 安全模型。其基本思想是:基于模块化观点,首先把 AKE 协议定义在理想模型中,由于理想模型相对简单(敌手是被动的)易于证明安全性,最后通过特定的“认证器”把协议“编译”成现实模型中的协议。文献[14]则在文献[12]、[13]的基础上,结合使用两种方法论,提出了适用于 PKI 环境中的 mBJM 安全模型;并针对协议结束时需要 Hash 函数处理才输出会话密钥的 AKE 协议(Hash 处理方式事实上已经在密钥交换协议设计中被广泛接受,如各类密钥导出函数),借助 Gap 假设,给出了较文献[13]更具有灵活性的模块化证明思想。

下面首先简单描述一下 mBJM 模型。该模型包括的每个用户 U 具有对应的公、私钥

(PK_U,SK_U),通常使用 Oracle Π_U^i形式化表示用户 U 的第 i 个通信实例(即形式化表示 U 的第 i 次运行协议),这些 Oracle 对各类输入的消息(询问)依据协议规则给出相应的输出(回答)。任何 Π_U^i 只可能处于 3 种状态之一:未决状态、接受状态、拒绝状态。一旦处于接受状态,Π_U^i 应该具有:角色 $role_U^i$∈{发起方,应答方};对应伙伴(Partner)的 ID,记为 pid_U^i(即意定的通信方);会话 ID sid_U^i及会话密钥 sk_U^i。

敌手 E 被形式化为一个概率多项式时间(PPT)算法,E 完全控制信道,并通过提出以上 Oracle 询问的方法与各种 Oracle 交互。

下面通过一个在挑战者 C′和敌手 E 之间的 game 来形式化定义 mBJM-AK 的安全性。

Game 12.1

(1) C′运行密钥和参数生成算法为用户分配参数和公、私钥(E 不知道私钥)。

(2) E 可以提出多项式数量级的 Oracle 询问,由 C′给出回答。E 可以提出以下询问:

① Send(Π_U^i,M):E 向 Π_U^i 发送消息 M,C′根据协议给出模拟回答。

② Reveal(Π_U^i):E 要求获得 Π_U^i 持有的会话密钥 sk_U^i。

③ Corrupt(U):E 要求获得 U 的私钥。

如果 E 曾经提出过 Reveal 询问,称 E 已经揭示了 Oracle Π_U^i;如果 E 曾经提出过 Corrupt 询问,称 E 已经收买了相应的用户;如果 Π_U^i 的伙伴没有被揭示且 pid_U^i没有被收买,称该 Oracle 是新鲜的(Fresh)。

④ Test($\Pi_{U^*}^i$):在某一时刻,E 可以对某新鲜 Oracle$\Pi_{U^*}^i$ 提出 Test 询问,C′随机选择比特 b。如果 b 为 1,C′输出 $sk_{U^*}^i$,否则输出随机数。此后 E 可以继续提出前面的 Oracle 询问,但禁止揭示 $\Pi_{U^*}^i$ 及其伙伴 Oracle,也不能收买 $pid_{U^*}^i$。

(3) 敌手 E 输出对比特 b 的猜测值(0 或 1)。

如果 E 对比特 b 的猜测正确,就称敌手赢得了 mBJM game。

如果协议满足强相伴性,且任何 PPT 敌手赢得 mBJM game 的概率优势是可忽略的,就称 AKE 协议满足 mBJM-AK 安全性。这里 AKE 协议的强相伴性是指:对于任何两个非意定的通信方而言,任何 PPT 敌手不可能以不可忽略概率使得二者均处于接受状态,并持有相同的会话密钥。有关 mBJM 模型的细节可参见文献[14]。

文献[14]提出的 AKE 协议模块化证明技术适用于上述的 mBJM 模型。基本思路是:首先证明 AKE 协议满足所谓的强相伴性(Strong Partnering),然后证明与该协议相关的某协议在“高度约化”模型中是安全的,最后利用“Gap 假设”把约化模型中的“相关协议”的安全性证明“转化成”一般模型中 AKE 协议的安全性证明。对于如上的 AKE 协议而言,这里所谓的“相关协议”是指,最终的会话密钥是某数字串(即 Hash 函数的输入)。

2. 条件可否认性

FAKE 协议作为一种特殊类型的 AKE 协议,与普通 AKE 协议的区别在于:

(1) FAKE 协议规定客户 C 为协议发起方(请求服务商 S 提供服务),S 作为协议应答方。

(2) C 和 S 共享一个由 C 产生的公共随机承诺 h,当考虑 mBJM-AK 安全性和可否认性时,h 视为协议的公共随机输入,h 由带有秘密输入 k 的算法 KC 产生,即h=KCommit(k),k 由 C 秘密产生,称为会话证据。

(3) 可否认性，在会话证据不被揭示的前提条件下，协议满足可否认性。

(4) 公平性，一旦会话证据 k 被 C 揭示，则协议双方都不能否认曾经运行过协议。

采用可证明安全性理论中常用的模拟论断来定义可否认性，为了与一般的可否认性定义加以区分，称为条件可否认性。

定义 12.1(条件可否认性) 称公平认证密钥交换协议 FAKE 是条件可否认的，如果满足：对任何 PPT 敌手 A，在会话证据 k 保密的条件下，存在一个模拟算法 SIM_A，其输入与 A 完全一样，输出的模拟观察(Simulated View)和敌手从协议执行得到的真实观察是计算不可区分的。

除了承诺秘密输入保密的条件，基本思想和文献[13]是一致的。上述敌手不同于传统上考虑的外部敌手，注意定义 12.1 的主要目的是防止敌手(可能是外部敌手，也可能就是协议的某一方，如服务器 S)试图向第三方证明曾经发生会话，因此敌手的观察包括内部掷币状态、完整的交互记录，特别还有 A 通过执行协议得到的会话密钥 K(如果会话没结束，会话密钥标记为错误，不失一般性，不考虑这种情况)。注意必须考虑到对输出会话密钥的模拟，即不仅密钥交互会话过程本身是可模拟的，而且会话密钥取值本身也应该是模拟输出的一部分，只有这样才能避免攻击者特别是内部敌手具有证明某网络服务是否曾经发生的能力，因为会话过程和输出会话密钥的可模拟性将使得敌手的“证据”变得毫无意义。这是与当前许多可否认认证协议的主要区别之一，通常也是安全性证明的难点所在。

仍然可通过一个在挑战者 C′和敌手 E 之间的“game”来形式化定义条件可否认性。

Game 12.2

(1) C′运行密钥和参数生成算法为用户分配参数和公、私钥(E 不知道私钥)。

(2) 类似于 Game1，可以提出 Oracle 询问 Send(Π_U^i, M)、Reveal(Π_U^i)、Corrupt(U)。

(3) C′随机选择某用户 $\Pi_{U^*}^i$ 作为模拟对象；但不允许敌手 E 同时对 U^* 及其对应伙伴 $pid_{U^*}^i$ 提出 Corrupt 询问，但允许询问其中之一，如果询问，由 C′向 E 提供对应的私钥，作为 E 的输入之一。

(4) C′构造一个模拟器 SIM，其输入与 E 完全一致。最后 C′运行 SIM 算法，输出会话记录和对最终会话密钥 $sk_{U^*}^i$ 的模拟值。

显然定义 12.1 是说，如果任何 PPT 敌手能够在多项式时间内区分如上的模拟观察和实际观察，那么就称 E 赢得 game12.2。否则就称协议满足条件可否认性。

3. 公平性

直观上来看，一个“公平”的认证密钥交换协议至少应该满足：只有产生会话证据的一方(客户 C)才可能揭示会话证据；在揭示会话证据的条件下，各方均不再满足可否认性，这时会话记录已经与各方身份“绑定”。

为了给出公平性的形式化定义，先考虑敌手 A 和挑战者 C′之间的以下“game”。

Game 12.3

(1) 初始化。主要是挑战者 C′运行密钥和参数生成算法，为用户分配公、私钥和相应参数。

(2) Oracle 询问。A 可以向挑战方做以下询问：

① Send(Π_U^i, M)：A 向 oracle Π_U^i 发送消息 M，后者根据协议给出回答。

② Reveal(Π_U^i)：A 请求揭示 Π_U^i持有的会话密钥。

③ Corrupt(U)：A 请求揭示 U 的私钥。

④ KCommit 询问：A 请求客户 C 随机选择 k，输出值 h=KCommit(k)；如果 A 愿意，A 也可以自己随机选择 k，计算 h=KCommit(k)。

⑤ KReveal 询问：A 请求客户 C 揭示以前的 KCommit 询问中的会话证据 k。

(3) 输出。最后敌手 A 输出涉及用户 X_c、X_d 的协议记录(k,h,T)，这里 k 是会话证据，h=KCommit(k)是公共随机承诺，T 是完整的协议记录(协议双方的收、发消息)。如果用户 X_c、X_d 至多有一个曾经被提出过 Corrupt 询问，且以下 2 种情况之一发生，称 A 赢得该 game：

① h 是以前的某 KCommit 询问的输出，但从没有过对应的 KReveal 询问。

② A 同时还输出(h,T')，这里新的协议记录 $T' \neq T$，T，T'是计算不可区分的。

定义 12.2(公平性) 称 FAKE 协议是公平的，如果任何 PPT 敌手赢得以上 game 的概率都是可忽略的。

以上定义是对“公平性”直观理解的形式化归纳和抽象。首先，协议对客户 C 是“公平”的，因为上述 game 输出的情况(1)确保只有产生会话证据的实体(客户 C)才可能揭示它，这样使得 C 产生的协议记录与自己的身份得以绑定。另一方面，情况(2)则保证了：任何一方(即使是“合法”通信方之一)都不能基于相同的会话证据再产生一份新的合法协议记录(即不再具有否认性)，即使是产生会话证据的客户 C 本身(当然应答方 S 就更是如此)。

应该特别指出的是，定义 12.2 的情形(2)与文献[11]中给出的并发签名的公平性定义是完全不同的。首先，文献[11]中的公平性定义只针对数字签名，而定义 12.2 是针对 AKE 协议；另外，文献[11]中有关并发签名的公平性定义也存在一定问题，它事实上只能确保 Keystone (相当于本文所称的会话证据)是由产生它的一方揭示，容易验证文献[11]中有关公平性 game 输出的情形(2)(主要是有关针对收方的公平性情形)囊括的情况集合必然是空集，因此导致文献[11]中的公平性定义的情形(2)没有说明任何问题。

12.6.3 一个具体的 FAKE 协议

本小节利用 8.12 节介绍的“并发签名”机制[11]设计了一个可证明安全的 FAKE 协议实例。

参数设置：公开大素数 p 和 q，满足 $q|(p-1)$，g 是 Z_p 的一个 q 阶生成元，$G=<g>\subset Z_p^*$，Hash 函数 $H:\{0,1\}^*\to\{0,1\}^l$，l 是安全参数。客户 C 的密钥对为($x_C\in_R Z_q^*$，$X_C=g^{x_C}$ mod p)，服务器 S 的密钥对为($x_S\in_R Z_q^*$，$X_S=g^{x_S}$ mod p)。

FAKE 协议执行过程如下：

(1) 发起方 C(客户)初始化某会话进程 Π_C^i，取定对应伙伴(服务器 S)的 ID，记为pid_C^i=S，随机选取临时参数 $c\in_R Z_q$，C 发送消息 $T_C=g^c$ mod p 给收方 S。

(2) 接到消息后，应答方 S 同样初始化某会话进程 Π_S^j，类似地，取定 pid_S^j=C；然后随机选取临时参数 $b\in_R Z_q$，发送消息 $T_S=g^b$ mod p 给 C。

(3) C 随机选取临时参数 $k\in_R Z_q$，计算 h=KCommit(k)，即 $h=H_1(k)$，计算并发送 $\sigma_C=\mathrm{CSig}_{x_C,h_2}(T_C,T_S)=(s_C,h_C,h)$给 S，这里 $\mathrm{CSig}_{x_S,h_2}(T_C,T_S)$表示以 h 为输入，对消息(T_C,T_S)的并发签名。

(4) 应答方 S 执行 NTVerify 算法，验证并发签名 σ_C 的合法性，如果合法，设置本次会话的 ID $\mathrm{sid}_C^i = X_C, X_S, T_C, T_S$，同样以询问值 h 为输入，计算对消息 (T_S, T_C) 的并发签名 $\sigma_S = (s_S, h_S, h)$，并发送该签名给 C。

FAKE 协议输出：

(1) 客户 C：验证收到并发签名 σ_S 的第 3 个签名分量是否是 $h = H_1(k)$，如果是，执行 NTVerify 算法，验证 σ_S 的合法性，如果合法，设置本次会话的 ID $\mathrm{sid}_C^i = X_C, X_S, T_C, T_S$，计算会话密钥 $K_C = H(T_S^{x_C} \| X_S^c \| \mathrm{sid}_C^i)$。

(2) 服务器 S：计算会话密钥 $K_S = H(X_C^b \| T_C^{x_S} \| \mathrm{sid}_S^i)$。

如果双方都处于接受状态，则客户 C 通过会话密钥 K_C 要求服务器 S 提供特定服务，而服务器 S 则使用会话密钥 K_S 来为客户提供服务。

12.6.4 FAKE 协议的安全性证明

首先 FAKE 协议满足正确性，因为如果不存在外来干扰，易见

$$K_C = H(T_S^{x_C} \| X_S^c \| \mathrm{sid}_C^i) = H(g^{bx_C} \| g^{cx_S} \| \mathrm{sid}_C^i) = H(g^{bx_C} \| g^{cx_S} \| \mathrm{sid}_S^i) = K_S$$

正如文献[11]中所指出的那样，为了使协议在 mBJM 模型中满足强相伴性，只需保证：每一会话的 ID 都是唯一的，且除了可忽略概率外，仅当 $\mathrm{role}_U^i \neq \mathrm{role}_{U'}^i$，$\mathrm{sid}_U^i = \mathrm{sid}_{U'}^i$，$\mathrm{pid}_U^i = U'$，$\mathrm{pid}_{U'}^i = U$ 同时成立，才满足 $\mathrm{sk}_U^i = \mathrm{sk}_{U'}^i$。可以看到，FAKE 的会话密钥是经 Hash 处理得到的，而 Hash 函数的输入字符串包含了 sid_U^i、U、U'、pid_U^i等必要的伙伴信息，特别是注意到随机选取的 T_C 和 T_S 均包含在会话 IDsid_U^i当中，因此易于在 ROM 中证明 FAKE 协议满足强相伴性。

FAKE 协议实际上是对文献[11]中的 AKE 协议的修改：主要加入了生成并发签名的步骤，但在考虑 mBJM-AK 安全性时，CSig 可以视为普通的数字签名，由于这是一个可证明安全的签名算法，因此协议的安全性证明和文献[11]中的相关证明没有本质差别，可直接套用文献[11]中提出的模块化证明技术，限于篇幅，这里不再赘述。

还应指出的是，应用文献[11]中的模块化证明技术时，必须应用由文献[15]中提出的 Gap 假设(下面给出的 GDH 问题是 Gap 假设的一个实例)。通俗地讲，Gap 问题就是这样一种问题，在决策 Oracle(如下文的 DDH Oracle)的帮助下，解决某计算问题(如 CDH 问题)。目前 Gap 问题已经在可证明安全领域获得了很多成功应用，也被同行广泛接受。这里，FAKE 协议的安全性证明主要基于以下问题。

(1) CDH 问题：已知 $g^a, g^b \in G(a, b \in_R Z_q)$，求解 $g^c = g^{ab} \bmod p$。

(2) DDH 问题：已知 $g^a, g^b, g^c \in G(a, b \in_R Z_q)$，判别 $c = ab$ 是否成立。

(3) GDH 问题：已知 $g^a, g^b \in G(a, b \in_R Z_q)$，同时已知一个可以解决 DDH 问题的 Oracle，求解 $g^c = g^{ab} \bmod p$。

如果求解以上问题的任何 PPT 算法的成功概率是可忽略的，就称求解以上问题是困难的。

定理 12.1 在 GDH 问题难解的意义下，FAKE 协议是 mBJM-AK 安全的。

限于篇幅，不再给出证明细节，基本方法与文献[11]中的方法是一致的。

定理 12.2 如果求解 GDH 问题是困难的，则 FAKE 协议满足条件可否认性。

证明 为了证明FAKE协议针对某恶意通信方的否认性，只需构造一个模拟器SIM与该恶意通信方模拟交互，使得模拟协议记录与该恶意方和真实方执行协议得到的协议记录是计算不可区分的。不失一般性，假设我们是模拟客户C(反过来也一样)与敌手A交互，根据形式化安全模型规划，SIM的输入与敌手完全一致，包括敌手运行密钥生成算法得到的收方S的公私钥对($x_S \in_R Z_q^*$，$X_S = g^{x_S} \bmod p$)，但C的私钥则是未知的。

首先，协议的步骤(1)、(2)与实际协议完全一样，即SIM初始化某会话进程Π_C^i，取定$\mathrm{pid}_C^i = S$，随机选取$c \in_R Z_q$，发送消息T_C给收方S，并接收来自S的消息T_S。至此敌手得到的对协议的模拟观察(View)和实际观察的分布显然是完全一致的。

由于并发签名CSig满足非传递性，即以(X_C，$X_S = g^{x_S} \bmod p$，x_S，(T_C，T_S))为输入，存在一个PPT算法FakeNTsign，输出一个合法签名(能够通过NTVerify算法验证)，该签名是与实际签名计算不可区分的。SIM运行上述算法：随机选择t，$h_C' \in Z_q$，计算$z' = g^t X_C^{h_C'} \bmod p$，$h = H_2(X_C \| X_d \| T_C \| T_S \| z')$，$h' = (h - h_C') \bmod q$，$s_C' = t - x_S h' \bmod q$，输出签名$\sigma_C' = (s_C', h_C', h')$，易于验证该签名可以通过NTVerify算法验证，并且和客户C的实际签名是计算不可区分的。发送该签名给S。

如果接收到签名$\sigma_S = \mathrm{CSig}_{x_S, h}(T_S, T_C) = (s_S, h_S, h)$，SIM验证$h = h'$是否成立以及签名是否合法，如果合法，设置本次会话的ID $\mathrm{sid}_C^i = X_C, X_S, T_C, T_S$。

由于CSig满足非传递性，因此敌手得到的模拟观察和实际观察是计算不可区分的。下面只需模拟协议的输出即会话密钥取值$K_C = H(T_S^{x_C} \| X_S^c \| \mathrm{sid}_C^i) = H(g^{bx_C} \| g^{cx_S} \| \mathrm{sid}_C^i) = H(g^{bx_C} \| g^{cx_S} \| \mathrm{sid}_S^j) = K_S$。在ROM(随机预言模型)中考虑这个问题，即H是随机预言。SIM维持一张表$L_H = \{(g^b, g^{x_C}, *, g^{cx_S}, \mathrm{sid}_C^i, K)\}$(因为已知S的私钥，因此计算$g^{cx_S}$是容易的)，开始为空。协议执行完毕，SIM随机选择$K \in \{0,1\}^l$，输出K，并添入表中对应位置。如果敌手A向Oracle H询问$g^{bx_C} \| g^{cx_S} \| \mathrm{sid}_S^j$，SIM首先检查会话IDsid$_S^j = X_C, X_S, T_C, T_S$是否出现在$L_H$中对应位置，如果没有，给予新的随机回答，并在表中加以标记。如果出现了，在DDH Oracle的帮助下，判断第一个询问分量是否是g^{bx_C}，如果不是，给予新的随机回答，并加以特别标记；否则，把表中对应位置的$*$替换成g^{bx_C}，并用K作为回答。

综上所述，SIM向接收方(敌手A)完整模拟了协议发起方C的行为，特别是根据ROM方法论，敌手不经询问随机预言即输出会话密钥的概率是可忽略的，因此敌手得到的模拟观察和实际观察是计算不可区分的。

类似地，可以证明模拟接收方S的情况。

定理12.3 FAKE协议满足公平性。

证明 我们在ROM中给出证明，即H_1、H_2、H都是随机预言。假设协议不满足公平性，即经过协议模拟的观察(包括前面game12.2中各类Oracle询问)，存在PPT敌手A以不可忽略的概率输出某会话证据，即Keystone k以及完整的协议记录T。T必然包含2个通过NTVerify和CSVerify算法验证的合法并发签名对$\sigma_C = (s_C, h_C, h)$，$\sigma_S = (s_S, h_S, h)$，这里$h = H_1(k)$。首先考虑公平性game12.2的情形(1)，由于H_1是随机预言，任何敌手在没有向H_1询问过k的情况下输出满足$h = H_1(k)$的k的概率显然是可忽略的。因此，唯一的可能就是情形(2)以可观概率发生，即基于同样的$h = H_1(k)$，敌手A以不可忽略的概率同时还输出新的协议记录T'，这意味着产生了新的合法并发签名，这显然与并发签名的不可伪造性(定理12.1)是矛盾的，因此公平性game12.3的情形(1)、(2)的发生概率均是可忽

略的。

12.7 小结

在很多应用场景(如电子交易、电子选举和电子支付)中,通信双方的利益往往不一致。因此,在设计这些场景中所应用的安全协议时必须遵循公平性原则,即参与协议的任何一方在协议执行的任何阶段都不能处于一种相对有利的地位,应保证交易双方都不能通过损害对方利益而得到它不应得的利益。本章重点介绍了一些典型的公平交换协议。

目前像网上购物或通过登录特定网站获取服务代理商的服务这样的网上交易业务日益普遍,关于服务商是否提供过某特定服务这样的纠纷也逐渐增多。如何平衡保护个人隐私和确保网上交易公平之间的矛盾已经成为一个不容忽视的问题。我们也用很大篇幅介绍了文献[7]中提出的解决该问题的思想、理论和方案。另外,我们在这一领域也取得了一些其他研究成果,感兴趣的读者可参阅文献[16]～[21]。

参 考 文 献

[1] Goldwasser S, Micali A. Probabilistic encryption, Journal of Computer and Systems Science, 1984 (28): 270～299.

[2] Pedersen T P. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, Advances in Cryptology—CRYPTO'91, volume 576 of LNCS, 129～140. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1992.

[3] Chaum D. Elections with unconditionally secret ballots and disruptions equivalent to breaking RSA, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1988, 177～181.

[4] Blum M. Coin flipping by telephone:a protocol for solving impossible problems,Proceedings of the 24th IEEE Computer Conference, 133～137,1982.

[5] Salomaa A. Public-key cryptography,Berlin: Springer-verlag,1990.

[6] Yao A C. Protocols for security computations, Proceedings of the 23rd IEEE Symposium on the Foundations of Computer Science, 160～164,1982.

[7] 冯登国,陈卫东. 公平认证密钥交换协议的安全模型与模块化设计,中国科学 F 辑:信息科学,2009, 39(10):1055～1062.

[8] Mario. Deniable authentication and key exchange. 2006.

[9] Dwork C, Naor M, Sahai A. Concurrent zero-knowledge. Proc. of 30th sysposium on theory of computing(STOC),ACM press, 409～418, 1998.

[10] Krawczyk H. SKEME: a versatile secure key exchange mechanism for Internet. Proc. of 1996 IEEE Symposium on network and distributed system security(SNDSS), 114～127.

[11] Kudla J. Special signature scheme and key agreement protocols. Thesis submitted to the university of London for the degree of doctor of philosophy. Information security group department of mathematics Royal Hollway, university of London,2006.

[12] Bellare M, Rogaway P. Entity authentication and key exchange. In: Stinson D. R, ed. Proc. of the Advances in Cryptology—Crypto'93. LNCS 773, Berlin, Heidelberg: Springer-Verlag, 232～249, 1993.

[13] Bellare M, Canetti R, Krawczyk H. A mod ular approach to the design and analysis of authentication and key exchange protocols. In: Proc. of the 30th Annual Symp. on the Theory of Computing. New York: ACM Preee, 1998. 419 ~ 428. http://doi. acm. org/10. 1145/276698. 276854.

[14] Kudla K, Paterson K G. Modular security proofs for key agreement protocols. In: Advances in Cryptology-ASIACRYPT 2005,Bimal Roy Ed. Berlin: Springer. 549～565,2005.

[15] Okamoto T, Pointcheval D. The gap-problems: a new class of problems for the security of cryptographic schemes. In K. Kim, editor, Public Key Cryptography-PKC 2001, volume 1992 of Lecture Notes in Computer Science, 104～118. Springer-Verlag,2001.

[16] Zhang Z F, Xu J, Feng D G. Efficient Identity-Based Protocol for Fair Certified E-mail Delivery. CANS, 2005: 200～210.

[17] Zhang Z F, Feng D G, Xu J, Zhou Y B: Efficient ID-Based Optimistic Fair Exchange with Provable Security. ICICS, 2005: 14～26.

[18] Zhang Z F, Feng D G. Efficient Fair Certified E-Mail Delivery Based on RSA. ISPA Workshops 2005: 368～377

[19] 徐静,张振峰,冯登国. 利用代理签名构造基于身份的优化公平交换协议(英文),软件学报,2007,18(03): 746～754.

[20] Jiang S Q, Feng D G, Qing S H. Analysis and Design of E-voting Protocol. SEC, 2000: 281～290.

[21] Zhang Z F, Feng D G, Xu J, Zhou Y B. Efficient ID-Based Optimistic Fair Exchange with Provable Security. ICICS, 2005: 14～26.

第 4 篇　应用安全协议

应用安全协议与具体应用有关，是用基础安全协议或密码算法结合具体实际应用构建的安全协议，有时也称安全方案。本篇主要介绍 Kerberos 协议、X. 509 协议、IPSec 协议、TLS/SSL 协议、入侵容忍 CA 协议、基于身份的 PKI 协议和可信计算平台远程证明协议等。

第13章　典型的分布式认证协议和网络安全通信协议

目前常用的分布式认证协议主要有两类：一类是基于对称密码算法的，典型代表是Kerberos协议；另一类是基于公钥密码算法的，典型代表是X.509协议。关于这两个协议中的密钥协商协议已在前面作了介绍，本章重点从应用的角度介绍这两个典型协议。IPSec协议和TLS协议是两个重要的网络安全通信协议。IPSec协议建立在IP层，它引入了数据认证机制、加密机制和相关的密钥管理，实现了较为全面的网络层安全。SSL协议建立在TCP协议栈的传输层，用于保护面向连接的TCP通信。

13.1　Kerberos协议

Kerberos协议是麻省理工学院(MIT)“雅典娜计划”(Project Athena)的一部分，它是MIT于20世纪80年代后期、90年代初期开发出来的系列基于对称密码算法的安全协议，主要应用于无安全措施的工作站、中等安全度的服务器以及高强度安全的密钥协商分布式环境中。对于运行于同一机器A的两个应用X和Y来说，Y仅仅要求A给出X的用户标识符(ID)。因为Y确信本地机器是安全的，如果A给出的X的用户标识符是Y所期望的，则X一定是X。但当X运行于不同的机器B中时，应用Y被强迫信任B能提供一个正确的回答。遗憾的是，在网络上要Y信任B是很困难的。因为所谓网络“黑客”十分容易假冒B，产生假的应答消息。Kerberos协议则不需要这种信任，而且X相信通过Kerberos协议能够给Y足够的信息来认证自己，Y不再需要信任机器B来认证X。同时，Kerberos协议还产生应用于X和Y之间进行保密通信的会话密钥，可进一步增强网络应用的安全性。

13.1.1　Kerberos协议结构

Kerberos协议由一个认证服务器(AS)和若干个(一个或多个)票据分配服务器(TGS)组成，其基本结构如图13.1所示。

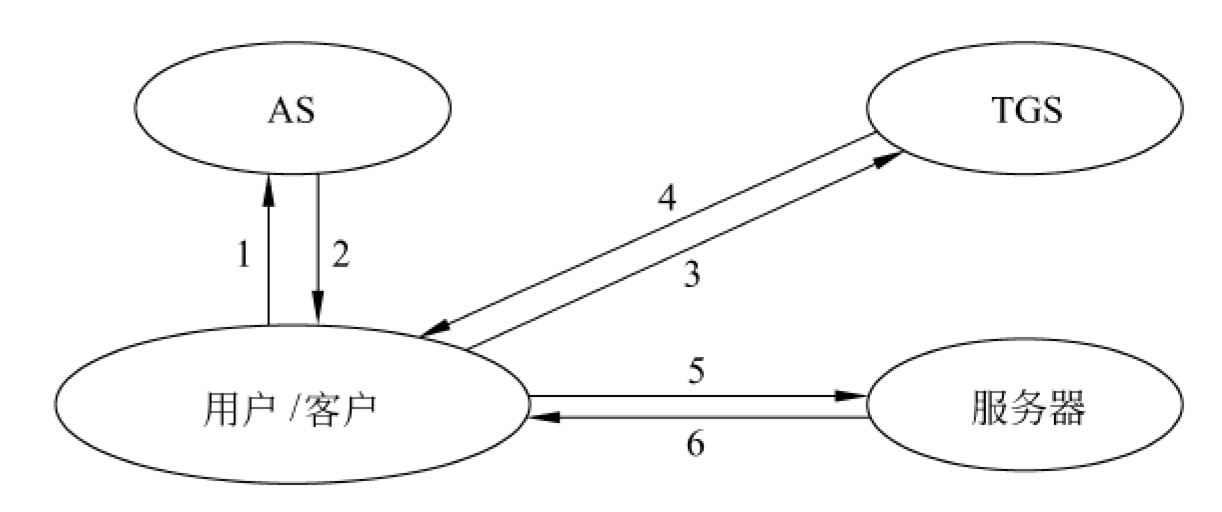


图13.1　Kerberos协议结构

Kerberos协议提供了一种主体验证与标识的方法，它的实现不依赖于主机的操作系

统,不依赖于主体地址的可信,也不依赖于网上所有主机的物理安全。它所基于的环境假设是:网上的通信包可被读取、修改及任意插入。

一个基本的 Kerberos 协议的认证过程如下。

(1) 客户请求认证服务器 AS 发给接入票据分配服务器 TGS 的票据。

(2) 认证服务器 AS 在其数据库中查找客户实体(注册的每个客户实体都与 AS 共享一个秘密密钥),产生一个会话密钥,AS 使用客户的秘密密钥对此会话密钥进行加密;然后生成一个 TGT(票据分配许可证),此许可证包括客户实体名、地址、TGS 名、时间戳、生存期、会话密钥等信息;并用 TGS 的秘密密钥对 TGT 进行加密;AS 把这两个加密结果发回给客户。

(3) 客户将第一个报文解密得到会话密钥,然后生成一个认证单,包括客户实体名、地址及时间戳,并用会话密钥对认证单进行加密。然后,向 TGS 发出请求,申请接入某一目标服务器的票据。此请求包括目标服务器名称、AS 发来的加过密的 TGT 及加密的认证单。

(4) TGS 用其秘密密钥对 TGT 进行解密,使用 TGT 中的会话密钥对认证单进行解密。然后将认证单中的信息与 TGT 中的信息进行比较。此时,TGS 产生新的会话密钥供客户实体与目标服务器使用,利用客户实体和 TGS 用的会话密钥对新的会话密钥加密;将新的会话密钥加入客户向该服务器提交的有效票据之中,票据中还包括客户实体名、网络地址、服务器名、时间戳、生存期等,并用目标服务器的秘密密钥将此票据加密;最后将这两个报文发回给客户。

(5) 客户将接收到的报文解密后,获得与目标服务器共用的会话密钥。这时,客户制作一个新的认证单,并用获得的会话密钥对该认证单进行加密。当请求进入访问目标服务器时,将加密的认证单和从 TGS 收到的票据一并发送给目标服务器。由于此认证单有会话密钥加密的明文信息,从而可证明发信人知道该密钥。

(6) 目标服务器对票据和认证单进行解密检查,包括地址、时间戳、生存期等。如果一切都核对无误,服务器则知道了客户实体的身份,并与之共享一个可用于它们之间的秘密通信的加密密钥。

在 Kerberos 协议中,客户向认证服务器 AS 请求证书有两种基本方法。第一种方法是,客户向 AS 发送为某一特定服务器请求票据的一个明文请求,请求用客户的密钥加密。通常这是请求认证服务器 AS 授权服务器 TGS 使用的 TGT。第二种方法是,客户向 TGS 发送一个请求,客户使用 TGT 来向 TGS 认证其身份。应答用 TGT 中的会话密钥加密。协议中将 AS 和 TGS 描述为不同的服务器,在实际执行中,它们是一个 Kerberos 服务器内不同的协议入口点。一旦获得,证书可用于验证传输中主体的标识以确保主体间传递消息的完整性,或消息的机密性。

为验证传输中主体的标识,客户将票据发送给应用服务器。由于票据是明文发送的且有可能为攻击者所截获或重用,因此同时发送一些附加消息用于证明此消息的确是来自于发布票据的主体。这些消息用会话密钥加密,并包括一个时间戳。时间戳证明了此消息是最近生成的且不能被重放。加密则证明了此消息是由一个拥有会话密钥的主体生成的。使用会话密钥还可保证主体间所交换的消息的完整性。此方法同时还可提供检测重放攻击和消息流修改攻击。具体做法是生成一个客户消息的摘要,并用会话密钥加密之后发送。

13.1.2 票据标志使用与请求

每个 Kerberos 票据包括一个用于指示其性质的标志集。票据的许多标志是客户所需要的，但有些标志是由所要求的 Kerberos 服务自动开启与关闭的。

(1) 初始、预认证以及硬件认证票据。INITIAL 标志表示一个票据是使用 AS 协议发布的，而不是基于 TGT 发布的。要求客户密钥证明知识的应用服务器可强调在它们所接收的任一个票据中设置此标志，并以此保证客户的密钥是最近分配给应用客户的。PRE-AUTHENT 和 HW-AUTHENT 标志提供了关于初始认证的附加信息，忽略了当前票据是直接发布的或是基于 TGT 发布的。

(2) 无效票据。INVALID 标志表示一个票据是无效的。应用服务器必须拒绝具有此标志的票据。一个过期的票据常常以此形式出现。无效票据必须是密钥分配中心(KDC)以前有效使用过的，VALIDATE 选项说明可在 TGS 的请求中提供给 KDC。KDC 只有在票据启用后才视其为有效。

(3) 可更新票据。应用将倾向于持有可长期有效的票据。然而这也使得其证书长期暴露于攻击者面前，而被窃的证书在票据有效期之前都将是有效的。仅使用短期票据和定期地获得新的票据将要求客户具有对其秘密密钥的长期访问，这带来的问题将更加严重。可更新票据的使用可降低票据被窃取的可能性。可更新票据具有两个有效期：一个是当前的票据期满时间；另一个是最近期满时间的允许值。一个应用客户必须定期向 KDC 提供一个可更新票据，并在 KDC 请求中设置 RENEW 选项。KDC 将发布一个具有一个新的会话密钥和一个更迟的期满时间的新票据。票据的其他域在更新过程中没有改动。当最近可允许的期满时间到达时，票据将永久期满。在每次更新过程中，KDC 将商议一个主机链用于确定其最近更新后票据是否被窃取过，KDC 将拒绝更新被窃取过的票据，因此被窃取票据的有效期将被缩短。

(4) 迟日期票据。有时应用会申请一个较后使用的票据，迟日期票据提供了在工作提交时从 KDC 获得此票据但直到 KDC 的一个未来请求激活之时才交付的获取票据的一种方法。如果在此期间一个票据被窃取过，那么 KDC 将拒绝使此票据有效，以此来挫败窃取行为。POSTDATED 标志表示一个票据是迟日期票据。应用服务器可检查票据中的认证时间域以判断原始的认证是何时发生的。有些服务将选择拒绝迟日期票据，或者它们只在原始认证开始后某一时间内接受迟日期票据。当 KDC 发布了 POSTDATED 票据，它也将被标记为 INVALID，因此应用客户必须向 KDC 提供在使用前有效的票据。

(5) 代理票据。有时一个主体必须允许一个服务以其名义来执行一个操作。在某些特定目的下，服务必须能够具有客户的标识，这可通过向其授权一个代理来实现。用代理标志来授权一个代理的过程被用于为特定服务的使用提供证书。票据中的 PROXIABLE 标志通常仅由票据授权的服务来解释。它可为应用服务器所忽略。当设置此标志时，等价于告诉票据授权服务可以发布一个基于此票据的具有一个不同的网络地址的新票据。此标志允许一个客户向一个服务器传递一个代理以其名义来执行一个远程请求。为了使被窃证书的使用更为复杂，Kerberos 票据只在票据中说明的网络地址上是有效的。当授权一个代理时，客户必须说明代理所使用的新的网络地址，或指明发布代理的任一个地址。

(6) 匿名票据。当政策允许时，为了在客户和服务器之间的加密通信中使服务器不能

够识别出客户,KDC 可发布匿名票据。匿名票据使用 KDC 配置的通用主体名来发布并设置 ANONYMOUS 标志。一个服务器接收此票据可要求后来的请求使用同样的票据以及使用原发自同一个用户的会话密钥。如果一个客户请求匿名通信,那么客户应当检查结果票据的确是匿名的。

(7) 转换策略检查票据。在 Kerberos 协议中,应用服务器对接受或拒绝认证负最终的责任,并且检查只有合适的可信 KDC 才能支持一个主体的认证。票据中的转换域标识在认证过程中涉及哪一个 KDC,那么一个应用服务器应当常规检查此域。终端服务器最终决定认证是否是有效的,其有效性是通过终端服务器的域应用一个确认转换域的域说明策略,并接受跨域认证证书来判断的。当 KDC 应用此检查并接受此跨域认证时,它将在其基于跨域 TGT 发布的服务票据中置 TRANSITED-POLICY-CHECKED 标志。

13.1.3 消息交换

1. 认证服务交换

当希望为一个已知的当前未持有证书的服务获取认证证书时,客户要发起一个客户和 Kerberos 认证服务器(AS)之间进行的认证服务交换(简称 AS 交换)。在其基本模式中,客户的秘密密钥用于加密和解密。此交换用于为 TGS 获得证书的登录会话开始之时,所获得的证书将用于随后的其他服务的证书的获得,且不必要求更多地使用客户的秘密密钥。

AS 交换包括两个消息:从客户到 Kerberos 的 KRB_AS_REQ,以及作为应答的 KRB_AS_REP 或者 KRB_ERROR。作为响应,客户发送了它自己的标识以及它将要申请的证书的服务器的名字。响应 KRB_AS_REP 包括一个由客户提供给服务器的票据,以及一个由客户和服务器共享的票据。会话密钥和附加的信息用客户的秘密密钥加密。KRB_AS_REP 消息包括用于检测响应的信息,并将之与它答复的消息相关联。

在没有预认证的情况下,认证服务不知道客户是否真正是请求中命名的主体,它仅发出一个回答而不关乎其他。这是可以接收的,因为只有名字标识在请求中的主体能够使用答复。其关键信息用主体的密钥加密。

(1) KRB_AS_REQ 消息的生成。客户将在其初始请求中说明一系列的选项,包括预认证是否进行,请求票据是否更新和代理,是否为迟日期票据,客户是否请求一个匿名票据等。客户准备 KRB_AS_REQ 消息并发送给 KDC。

(2) KRB_AS_REQ 消息的接收。如果一切运行正常,KRB_AS_REQ 消息的处理将导致客户提供给服务器一个票据的生成。

(3) KRB_AS_REP 消息的生成。认证服务器在其数据库中查找 KRB_AS_REQ 中命名的客户和服务器,分别提取其密钥。如果请求客户在请求中的主体命名由于不在认证服务器的数据库中而不可知,那么将返回一个出错信息 KDC_ERR_C_PRINCIPAL_UNKNOWN。

(4) KRB_ERROR 消息的生成。多种错误有可能发生,AS 通过返回出错消息 KBB_ERROR 给客户来表示不同的错误类型。

(5) KRB_AS_REP 消息的接收。如果响应的消息类型是 KRB_AS_REP,那么客户将验证应答的明文部分中 cname 和 crealm 域与其请求是否是匹配的。客户用其密钥解密应答的加密部分,验证加密部分的随机数与它请求中提供的随机数是否是匹配的。同时它还

验证响应中的 sname 和 srealm 与请求中的相关部分是否匹配。之后，客户存储此票据、会话密钥、起始和期满时间以及稍后用到的信息。

2．客户/服务器认证交换

客户/服务器(CS)认证交换用于网络应用对客户和服务器之间的相互认证。客户必须使用 AS 或 TGS 交换已为服务获得的证书。

客户/服务器认证交换包括两个消息，即 KRB_AP_REQ 消息和 KRB_AP_REP 消息。KRB_AP_REQ 消息包括一个认证传递的第一个消息的一部分认证消息，包括一个票据、一个鉴别码以及一些附加的簿记信息。票据自身是不足以认证一个客户的，这是因为票据在网上是以明文方式传播的，因此认证通过向服务器证明客户知道票据的会话密钥来阻止票据的无效重放。

(1) KRB_AP_REQ 消息的生成。当一个客户希望发起对一个服务器的认证时，它为想得到的服务获得一个票据和会话密钥。客户将重用它所持有的有效期内的票据。

(2) KRB_AP_REQ 消息的接收。认证是基于服务器的当前时间、鉴别码和票据的。有发生多种错误的可能。如果一个错误发生了，服务器将被认为回应给客户一个 KRB_ERROR 消息。如果它的“raw”形式不为协议所接受，则此消息将封装在应用协议中。KRB_ERROR 消息的格式将在后面给出描述。验证认证消息的算法如下：如果消息的类型不是 KRB_AP_REQ，服务器将返回 KRB_AP_ERR_MSG_TYPE 错误；如果 KRB_AP_REQ 中的票据所指示的密钥的版本服务器不能使用，将返回 KRB_AP_ERR_BADKEYVER 错误；如果设置了 USE_SESSION_KEY 标志，它向服务器指示票据是使用服务器 TGT 的会话密钥而不是其自己的会话密钥加密的。由于服务器可能在多个域内用不同的密钥注册，KRB_AP_REQ 中的票据的未加密部分的 srealm 域用于说明服务器解密票据所使用的密钥；如果服务器没有解密的正确密钥，将返回一个错误代码 KRB_AP_ERR_NOKEY。

(3) KRB_AP_REP 消息的生成。客户的请求将在同一个消息中包括认证信息和它的初始请求，而且服务无需明确回应 KRB_AP_REQ。然而，如果执行了双向认证，KRB_AP_REQ 消息将在其 ap-options 域中置 MUTUAL-REQUIRED，而且 KRB_AP_REP 消息将作为响应。

(4) KRB_AP_REP 消息的接收。如果收到一个 KRB_AP_REP 消息，客户使用从服务器获得的证书中得到的会话密钥解密消息，并验证时间戳等信息与它发给服务器的鉴别码中的相关域是否匹配。如果是匹配的，那么客户可确信服务器是诚实的。

在 KRB_AP_REQ/KRB_AP_REP 交换发生之后，客户和服务器共享一个可为应用所用的加密密钥。在有些情况下，此会话密钥的使用是在协议中隐含执行的。一个应用为随后的完整性和机密性保护来协商一个使用的密钥是客户在鉴别码的子密钥域中所建议的一个密钥。之后服务器将使用建议的密钥选择一个密钥，并在应用应答的子密钥域中返回一个新的子密钥。此密钥可用于随后的通信中。

3．TGS 交换

客户和 Kerberos 票据授权服务器 TGS 间的 TGS 交换是当客户希望为某一服务申请认证证书时，当它希望更新或使一个已有的票据有效时，或者当它希望获得一个代理票据时由客户发起的。在第一种情况下，客户必须已经为 TGS 申请了一个票据用于 AS 交换。

TGS 交换的消息格式与 AS 交换的消息格式是类似的，主要差别在于 TGS 交换中的加/解密不使用客户的密钥，而是使用来自票据授权票据(TGT)或可更新票据的会话密钥，或者来自鉴别码的子会话密钥。

TGS 交换包括两个消息：客户向 Kerberos 票据授权服务器 TGS 发送的一个请求 KRB_TGS_REQ 和一个对此消息的应答消息 KRB_TGS_REP 或者 KRB_ERROR。KRB_TGS_REQ 消息包括认证客户的信息和一个证书请求。认证信息包括认证头，其中有客户之前获得的票据授权、可更新的票据或无效票据。在 TGT 和代理的情况下，请求包括网络地址链、票据中封装的类型授权数据集或者附加的票据。TGS 的应答 KRB_TGS_REP 包括所请求的用会话密钥加密的证书。KRB_ERROR 消息包括一个错误代码和一个说明错误的文本。此消息是不加密的。

(1) KRB_TGS_REQ 消息的生成。在向票据授权服务器 TGS 发送一个请求之前，客户必须确定应用服务器注册的域。当客户拥有了适当域的 TGT 后，它将确定哪一个 Kerberos 服务器服务哪个域并与之联系。同 AS 交换一样，客户将在 KRB_TGS_REQ 消息中说明一个选项号，提供一个鉴别码头作为 padata 域的一个元素，包括在 KRB_AS_REQ 消息中使用的同样的域及一些选项域。

(2) KRB_TGS_REQ 消息的接收。KRB_TGS_REQ 消息与 KRB_AS_REQ 消息的处理过程是类似的，但仍有一些其他的检查需要进行。首先，Kerberos 服务器必须确定所伴随的票据所属的服务器并选择一个适当的密钥解密之。如果 TGT 是由另一个域发布的，那么必须使用正确的域内密钥。如果伴随的票据不是当前域的 TGT，而是当前域的一个应用服务器的 TGT，RENEW，WALIDATE 或 PROXY 选项在请求中进行了说明，并且票据请求的服务器是名字在伴随票据中的服务器，那么 KDC 将用它发布的服务器的密钥来解密认证头中的票据。之后，用户支持的认证头中的校验和必须进行验证。如果校验和的类型是不支持的，将返回一个 KDC_ERR_SUMTYPE_NOSUPP 消息。

(3) KRB_TGS_REP 消息的生成。KRB_TGS_REP 消息格式与 KRB_AS_REP 消息格式一样，只是其类型域置为 KRB_TGS_REP，包括一个被请求的服务器的票据或为获得被请求的票据而连接的一个中间 KDC 的票据授权服务器的票据。通过质询 Kerberos 数据库可找到正确服务器的记录。如果请求是要求一个远程域的 TGT 时，并且如果与所请求的域之间没有共享密钥，那么 Kerberos 服务器将选择一个它与之有共享密钥的，且与被请求的域最近的域并使用此域。

(4) KRB_TGS_REP 消息的接收。当客户收到 KRB_TGS_REP 时，处理方式与上述的 KRB_AS_REP 的处理方式相同。主要的区别是响应的密文部分必须使用来自 TGT 的会话密钥进行解密，而不是用客户的秘密密钥。

4. KRB_SAFE 交换

KRB_SAFE 消息可为客户用于申请检测对交换的消息修改的能力。

(1) KRB_SAFE 消息的生成。当一个应用欲发送一个 KRB_SAFE 消息时，它收集其数据和正确的控制信息并为之计算一个校验和，校验和使用子会话密钥或会话密钥来生成。KRB_SAFE 消息的控制信息包括时间戳和一个序列号。当所有发送的消息为对方接收时，序列号是有用的。如果一个应用程序期望忍受消息的丢失而不重发，那么时间戳是一个适当的重放检测机制。在计算校验和之后，客户将消息发送。

(2) KRB_SAFE 消息的接收。当一个应用接收了一个 KRB_SAFE 消息后，它做以下检验。如果出现了任何的错误，应用将记录下一个错误代码。首先检查的是消息中的协议版本号和类型域是否符合要求。如果不匹配则返回 KRB_AP_ERR_BADVERSION 或 KRB_AP_ERR_MSG_TYPE 错误。如果检查的校验和是不正确的，则返回一个错误信息 KRB_AP_ERR_INAPP_CKSUM。此外，如果消息中包含了一个不正确的序列号，则返回一个错误信息 KRB_AP_ERR_BADORDER；KRB_AP_ERR_MODIFIED 表示对收到信息计算的校验和与收到的检验和不匹配。

如果通过了所有的检查，那么应用可确定消息是由其通信对方发出，且在传递过程中没有被修改。

13.2　X.509 协议

OSI 目录检索服务标准 X.500 首次公布于 1988 年，该标准中包括了一部分陈述认证的标准，即 ISO/IEC 9594-8 或 ITU-T X.509 建议。根据分析和研究结果，1993 年和 1995 年分别对 X.509 建议作了微小修改，目前已发展到了第 4 版本。

13.2.1　X.509 协议结构

为了在成员 A 和 B 之间提供相互认证，选择双向认证或者三向认证，如图 13.2 所示。为了完成认证，A 需要 B 的公钥证书，而 B 需要 A 的公钥证书。为了核对证书，进一步证实公钥，也许需要一个证书授权链。

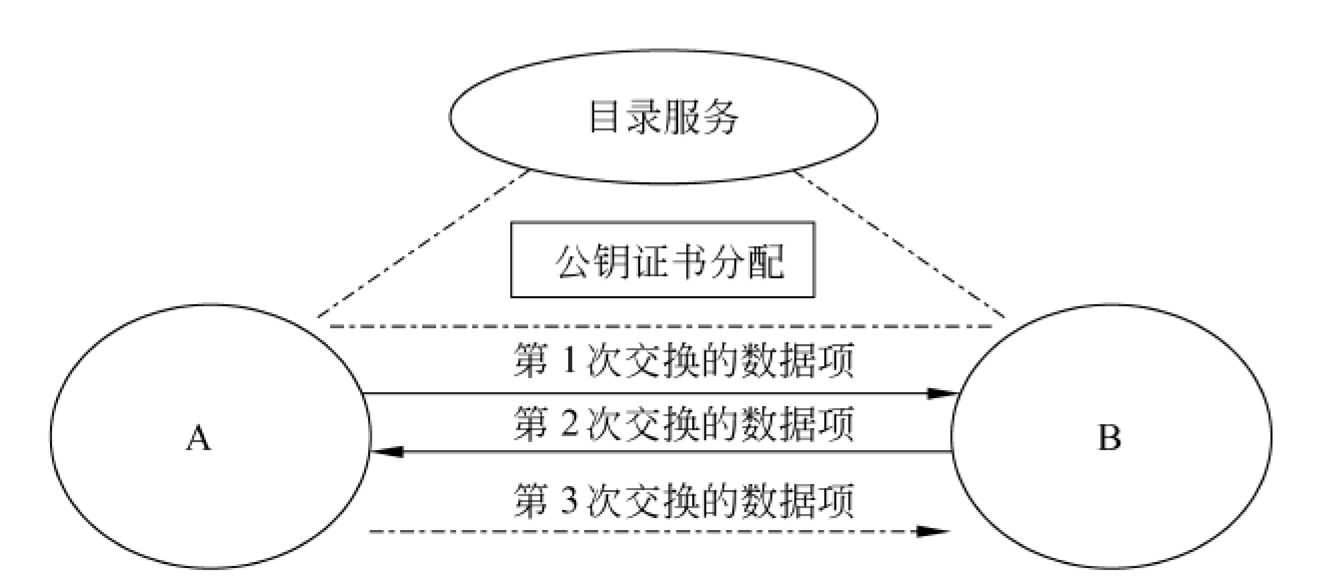


图 13.2　X.509 协议结构

公钥证书可通过访问目录服务或 A 和 B 之间通过认证直接交换证书来获得。

在描述 X.509 认证交换时，用到了以下记号：

$E_x\{\}$：表示在成员 x 的公钥下一系列数据值的加密结果。

$S_x\{\}$：表示一系列数据值连同用成员 x 的私钥对这些值进行的签名。

ts_{xy}：表示由成员 x 产生的一个当前时间戳，帮助成员 y 检查重放消息(也可能包含所传递消息的产生和过期日期/时间)。

nrv_{xy}：表示由成员 x 发送的一个非重复值，帮助成员 y 检查重放消息。

key_{xy}：表示由成员 x 产生的一个秘密密钥，用于保护 x 和 y 之间的通信。

在双向交换中，传递的数据项如下。

第 1 个数据项：A，$S_A\{ts_{AB}, nrv_{AB}, B, E_B\{Key_{AB}\}\}$。

第 2 个数据项：$S_B\{ts_{BA},nrv_{BA},A,nrv_{AB},E_A\{Key_{BA}\}\}$。

$E_B\{Key_{AB}\}$和 $E_A\{Key_{BA}\}$是可选择的。在第 1 个数据项的接收端，B 验证 A 的签名，检查消息中的 B 的识别符是否是正确的，检查时间戳是否是当前的，检查用于免遭重放的非重复值(如果使用了非重复值的话)。在第 2 个数据项的接收端，A 完成相应的一系列操作。值得注意的是，对每一个消息，签名验证提供了基本的认证，时间戳或非重复值使同一验证者免遭重放攻击，包含在消息中的接收者的识别符使不同验证者免遭重放攻击。以加密形式发送的 Key_{AB}或 Key_{BA}是可选择的。它们的用途没有具体说明，但通常人们将它们用于支持基于对称密码技术的数据的机密性或完整性保护。

在三向交换中，传递的前两项数据项除了不需要时间戳之外与双向交换中的一样。另一个从 A 传到 B 的数据项如下。

第 3 个数据项：$S_A\{B,nrv_{BA}\}$。

在三向交换中，在第 2 个数据项的接收端，成员 A 检查收到的 nrv_{AB}是否和在第 1 个数据项中发送来的相应值一样。类似地，在第 3 个数据项的接收端，成员 B 检查收到的 nrv_{BA}是否和在第 2 个数据项中发送来的相应值一样。B 也检查 A 对第 3 个数据项的签名。在三向交换中，用于重放检测目的的时间戳不再需要。

13.2.2 X.509 v3 证书

公钥的使用者应当确信加密或数字签名算法中使用的相关私钥为真实的远程主体(人或系统)所拥有，这是通过使用公钥证书来实现的。公钥证书是绑定公钥与主体的数据结构。绑定是通过可信的证书认证机构(CA)签署每个证书来实现的。

X.509 证书最为广泛的一个应用是在支持 SSL 协议的 Web 浏览器中的应用。其他基于 X.509 证书的技术有各种代码签名机制(如签名的 Java Archives、Microsoft 认证码)、各种安全电子邮件标准(如 PEM、S/MIME)和电子商务协议(如 SET)。

有两种基本的获得证书的方法：一种是主体自己建立；另一种是主体请求一个 CA 为其签发一个。

X.509 定义了证书中包含的信息，并给出了其格式。所有的 X.509 证书包括下列信息和一个签名：版本、序列号、签名算法标识、发布者名称、有效期、主体名称、主体公钥信息。

证书中的所有数据使用两个相关的标准进行编码，即 ASN.1 和 DER。ASN.1 描述数据，DER 描述了一个单一的存储和传递数据的方法。

1. 证书路径

如果公钥使用者没有持有一个由 CA 签名的证书，那么它将需要一个附加的证书以获得此公钥。一般而言，需要一个包括 CA 签署的公钥拥有者的证书和多个其他 CA 签署的证书的多级证书链，此证书链也称为证书路径。

2. 证书撤销

当一个证书被签发时，在其整个有效期内是可使用的。然而，多种情况可导致证书在其有效期期满之前变为无效。如名称的改变，主体与 CA 之间关联的改变，相应私钥的崩溃或有崩溃的嫌疑。在这些情况下，CA 需要撤销证书。X.509 定义了证书撤销的一种方法。这种方法包括每个 CA 定期发布的一个称为证书撤销链(CRL)的签名数据结构。CRL 是

标识由 CA 签署的并可从公共存储区中自由获得的撤销证书的一个时间戳链。每个撤销的证书以其证书序列号标识在 CRL 中。当系统用户使用一个证书时(如用于验证一个远程用户的数字签名),此系统不仅检查证书签名和有效性,还获得一个适当的近期 CRL,并检查此证书序列号不在 CRL 上。“适当的近期”的含义可能与本地策略不相同,但它往往指最近发布的 CRL。一个 CA 周期地发布一个新的 CRL,增加到 CRL 的一个条目作为跟随撤销通知的下一个更新,而当一个条目出现在一个超过撤销证书有效期后发布的 CRL 中时,将被从当前 CRL 中去掉。

这种撤销方法的优点是 CRL 可以使用与证书自身完全一样的方式进行分配,即通过不可信通信和服务器系统。使用不可信通信和服务器系统的一个局限性是,撤销的时间间隔将局限于 CRL 发布的周期。例如,如果一个撤销此时被记录,那么直到下一次 CRL 发布时撤销才被可靠地通知证书使用系统。此间隔取决于 CA 发布 CRL 的时间间隔。

13.2.3　证书及其扩展

证书的应用范围极其广泛,因而其互操作性要求也极强。

1. 基本证书域

这里给出 Internet 中使用的 X.509 V3 证书的描述。

(1) Certificate。Certificate 是一个包括以下 3 个域的序列。

① tbsCertificate。此域包括主体和发布者的名称、主体的公钥、一个有效期以及其他相关信息。

② signatureAlgorithm。此域包括 CA 用于签名此证书的密码算法的标识。

③ signatureValue。此域包括一个数字签名。为生成此签名,CA 证书证明 tbsCertificate 域中信息的有效性。特别地,CA 证明公钥与证书主体之间的绑定。

(2) TBSCertificate。TBSCertificate 是一个包括与证书主体和发布证书的 CA 相关的信息的序列。每个 TBSCertificate 包括主体和发布者的名称、主体的公钥、一个有效期、一个版本号、一个序列号和扩展。

① 版本。此域描述了编码证书的版本。

② 序列号。序列号是 CA 签发给每个证书的一个整数。对于每个证书它必须是唯一的。

③ 签名。此域包括 CA 用于签名证书的算法的标识。此标识必须与 Certificate 的 signatureAlgorithm 域中算法标识相同,而选项参数域的内容将根据不同的算法标识而有所不同。

④ 发布者。此域标识了签名和发布证书的主体,还必须包括一个非空可识别名(DN)。

⑤ 有效期。证书的有效期是证书保证其支持有关证书状况的信息的一个时间间隔。此域表示为 SEQUENCE,包括两个日期,即一个表示证书有效期开始的日期(notBefore)和一个表示证书有效期结束的日期(notAfter)。

⑥ 主体。此域标识与存储在主体公钥域内的公钥相关联的实体。

⑦ 主体公钥信息。此域包括公钥以及公钥所使用的算法的标识。算法用 AlgorithmIdentifier 结构来标识。

⑧ 唯一标识。此域出现在证书中用于处理主体/发布者名称的重用的可能。

⑨ 扩展。此域是以一个或多个证书扩展的一个 SEQUENCE 出现的。

2. 证书扩展

X. 509 V3 证书的扩展提供了关联用户或公钥的附加性质和管理验证层次的方法。X. 509 V3 证书格式允许一个团体定义私有的扩展,从而使此团体具备独有的信息。一个证书的每个扩展被指定为关键的或非关键的。当一个证书使用系统遇到一个它不能识别的关键扩展时它必须拒绝此证书,但如果是一个非关键扩展,尽管它无法识别也是可以忽略不计的。

(1) 标准扩展。这里给出 X. 509 定义的用于 Internet PKI 中的标准证书扩展。每个扩展与 X. 509 定义的一个 OID 相关联。

① 权威密钥标识。权威密钥标识扩展提供了识别用于签名一个证书的私钥所对应的一个公钥的一种方法。当一个发布者具有多个签名密钥时,则使用此扩展。识别必须基于公钥标识或基于发布者的名称和序列号。

② 主体密钥标识。主体密钥标识扩展提供了识别包括特定公钥证书的一种方法。主体密钥标识的值必须为权威密钥标识扩展的密钥标识域中的值。

③ 密钥使用。密钥使用扩展定义了证书中包含的密钥的作用。当一个密钥可用于多个操作中时将用此扩展约束。

④ 私钥使用期。私钥使用期扩展允许证书发布者为私钥说明一个有效期,主要是为签名密钥使用。此扩展包括两个选项组件:notBefore 和 notAfter。由两个组件说明的时间之前和之后与证书相关联的私钥不应用于签名。

⑤ 证书策略。证书策略扩展包括多个策略信息术语的一个系列,每个术语包括一个目标标识(OID)和可选择的语句。这些策略信息术语指出了证书发布所基于的策略以及证书使用的目的。有特定策略要求的应用将有一个由它们所接受的策略组成的策略链并将证书的策略 OIDs 与链中策略的 OIDs 进行比较。证书策略书写者和证书发布者使用两类策略语句:CPS Pointer 和 User Notice。前者包括一个指示器,用于指示由 CA 发布的证书实践声明 CPS。后者的目的在于当一个证书使用时显示给响应方。User Notice 包括两个可选择的域:noticeRef 和 explicitText。

⑥ 策略映射。策略映射扩展用于 CA 证书。它列了一个或多个 OID 对,每个对包括一个 issuerDomainPolicy 和 subjectDomainPolicy。

⑦ 主体可选名称。主体可选名称扩展可将附加的标识绑定到证书主体。定义的选项包括一个 Internet 电子邮件地址、一个 DNS 名称、一个 IP 地址和一个统一源标识(URI)。当这样的附加标识绑定到一个证书中时,必须使用主体可选名称扩展。

⑧ 发布者可选名称。与上述的主体可选名称扩展类似,发布者可选名称扩展用于关联 Internet 类型标识与证书发布者。

⑨ 基本约束。基本约束扩展标识证书的主体是否为一个 CA,以及贯穿此 CA 的一个证书路径的深度。pathLenConstraint 域只在 cA 为 True 时才有意义。此时,它给出了在一个证书路径中跟随此证书的 CA 证书的最大数目。0 值表示在路径中只有一个终端主体证书跟随其后。

⑩ 名称约束。名称约束扩展只在 CA 证书中使用,用于指出由一个证书路径中后来的证书的所有主体名称组成的一个名称空间。对于 URIs,约束应用于名称的主机部分,它将

说明一个主机或一个域，如“foo. bar. com” 和“. xyz. com”。当约束以一个句点开始时，它可被扩展为多个子域，如“abc. xyz. com”和“abc. def. xyz. com”都满足约束“. xyz. com”。然而，“xyz. com”是不满足约束“. xyz. com”的。当约束不是以一个句点开始时，它说明了一个主机。

⑪ 策略约束。策略约束扩展以两种方式约束路径的有效性。一种是禁止策略映射；另一种是要求路径中的每个证书包含一个可接受的策略标识。

⑫ CRL 分配点。该点扩展标识 CRL 信息是如何获得的。如果 cRLDistributionPoints 扩展包括一个类型为 URI 的 DistributionPointName，那么 URI 是当前 CRL 的一个指针且将由相关的 cRLIssuer 发布。

(2) 私有扩展。这部分定义了 Internet 公钥基础设施中使用的一个新的扩展。此扩展可用于识别支持发布 CA 的一个有效在线服务的直接应用中。由于信息可以由多种格式获得，每个扩展是一个表示 URI 的 IA5String 值的序列。URI 隐含地说明了信息及获得信息的位置和格式。一个对象标识与 id-pkix 名称空间内的 arc id-pe 下定义的私有扩展相关联。Internet PKI 未来定义的扩展也将是基于 arc id-pe 的。

授权信息访问扩展指出了如何访问证书发布者的 CA 信息及服务，信息及服务包括在线有效服务和 CA 策略数据。此扩展包含在主体或 CA 证书中。

AuthorityInfoAccessSyntax 序列中的每一项描述了 CA 发布包含此扩展的证书的附加信息的格式与位置。信息的类型和格式由 accessMethod 域来说明，accessLocation 域则说明了信息的位置，检索机制隐含在这两个域中。

RFC2459 文档为 accessMethod 定义了一个 OID，当附加信息列出的 CAs 发布的证书为发布包含此扩展的证书的 CA 的上级时，使用 id-ad-caIssuers OID。CA 发布者引用描述有助于证书使用者选择一个终止在为证书使用者信任的一点上的证书路径。当 id-ad-caIssuers 以 accessLocation 类型出现时，accessLocation 域描述了引用描述服务器和获取引用描述的访问协议。accessLocation 域定义为一个可以采用多种形式的 GeneralName。

13.2.4 CRL 及其扩展

RFC2459 定义了每个 CRL 中出现的信息的一个基准集，以及 CRL 内的常用性质的公用地址和这些性质的常用表达。

1. CRL 域

CertificateList 是一个至少包括以下 3 个域的序列：

(1) tbsCertList。此域是一个包括发布者名称、发布日期、下一个链的发布日期、撤销证书链以及可选的 CRL 扩展。而且，撤销证书链的每一项由使用者证书序列号、撤销日期以及可选 CRL 项扩展组成的序列来定义。

(2) signatureAlgorithm。此域包括 CA 用于签名 CertificateList 的算法的算法标识符。

(3) signatureValue。此域包括一个数字签名，ASN. 1 DER 编码的 tbsCertList 作为数字签名函数的一个输入，签名的结果也放入此域中。

2. CRL 扩展

CRL 扩展提供了一种关联附加性质与 CRL 的一种方法。X. 509 V2 CRL 格式也允许

主体定义为其携带独一无二的信息的私有扩展。一个 CRL 的每个扩展可设置为关键的或非关键的。如果一个 CRL 遇到一个关键性的扩展却不能处理,那么此 CRL 是无效的。然而,一个不可识别的非关键扩展将被忽略。

(1) 授权密钥标识。授权密钥标识扩展提供了一种识别与签名一个 CRL 的私钥对应的公钥方法。标识可基于密钥标识或基于发布者名称和序列号。当一个发布者拥有多个签名密钥时,此扩展是非常有用的。

(2) 发布者可选名称。发布者可选名称扩展可将附加标识与 CRL 的发布者相关联。定义的选项包括 rfc822 名称、DNS 名称、IP 地址及 URI。

(3) CRL Number。CRL Number 是一个非关键 CRL 扩展,它传达了 CA 发布的每个 CRL 的一个单调递增的序列号。此扩展允许用户易于确定何时一个特定的 CRL 替代另一个 CRL。

(4) Delta-CRL 指示器。Delta-CRL 指示器是一个关键 CRL 扩展,用于标识一个 Delta-CRL。通过以一个格式来替代 CRL 结构以存储撤销信息,Delta-CRL 可显著改善应用的处理时间。BaseCRLNumver 的值标识了作为生成此 Delta-CRL 的起点的基准 CRL 的 CRL 数。Delta-CRL 包括基准 CRL 和与 Delta-CRL 一同发布的当前 CRL 之间的变换。

发布分配点是一个关键 CRL 扩展,可标识一个特定 CRL 的 CRL 分配点,并且指出了 CRL 是否仅包括终端证书的撤销,或者 CA 证书的撤销,又或者是理由码的一个有限集。CRL 是用 CA 的私钥签名的,CRL 分配点没有自己的密钥对。与分配点关联的理由码在 onlySomeReasons 中说明,如果 onlySomeReasons 没有出现,分配点将包括所有理由点的撤销。CAs 可使用 CRL 分配点基于崩溃和常规撤销来划分 CRL。在这种情况下,理由码 keyCompromise(1)和 cACompromise(2)的撤销出现在一个分配点,其他理由码的撤销出现在另一个分配点中。

3. CRL 输入项扩展

CRL 输入项扩展提供了关联附加属性与 CRL 扩展的一种方法。CRL 输入项的每一个扩展可设置为关键的或非关键的。如果一个 CRL 遇到一个关键的 CRL 输入项扩展项却不知如何处理,那么此 CRL 将无效。然而,一个非关键的 CRL 输入项扩展将被忽略不计。下面将给出在 Internet CRL 输入项和信息标准位置内使用的推荐扩展的说明。

(1) 理由码(Reason Code)。理由码是一个非关键 CRL 输入项扩展,用于标识证书撤销的理由。CA 在其 CRL 输入项中包括有意义的理由码。

(2) 持有指令码(Hold Instruction Code)。持有指令码是一个可提供注册指令标识的非关键 CRL 输入项扩展,用于指出遇到一个已持有证书后所发生的行为。

(3) 无效日期。无效日期是一个非关键 CRL 输入项扩展,可提供一个已知的或猜测的私钥崩溃的日期或者证书无效的日期。此日期可早于 CRL 输入项中的 CA 处理撤销的撤销日期。当一个撤销由 CA 在一个 CRL 中首次登记时,无效日期可早于较早 CRLs 的发布日期。

(4) 证书发布者。此 CRL 输入项扩展标识了与一个间接的 CRL 的一个输入项相关联的证书的发布者。如果此扩展没有出现在一具非直接的 CRL 的第一个输入项中,证书发布者默认为 CRL 发布者。在一非直接的 CRL 的后继输入项中,如果此扩展没有出现,此输入项的证书发布者与先前的输入项的证书发布者一样。

13.2.5 证明路径的检验

证明路径处理用于验证对象可区别名称/对象可选名称和对象公钥之间的绑定。绑定由构成路径的证书中说明的约束所限制。

1. 基本路径检验

一个证明路径是一个 n 证书的序列，其中

(1) 对于所有的$\{1,2,\cdots,n-1\}$中的 x，证书 x 的对象是 $x+1$ 的发布者。

(2) 证书 $x=1$ 是一个自签名的证书。

(3) 证书 $x=n$ 是一个终端证书。

路径处理逻辑的输入包括：

(1) 一个长为 n 的证明路径。

(2) 一个初始策略标识集，它标识了一个或多个可为证明路径处理所接受的证书策略。

(3) 当前日期/时间。

(4) 确定路径有效性的时间 T。

从 $i=1$ 到 n 的每个证书的路径处理软件执行的行为描述如下：自签名证书是证书 $i=1$，终端证书是 $i=n$。处理是按序进行的，因此证书 i 的处理影响处理证书 $i+1$ 的状态变量。路径处理行为包括以下内容。

(1) 验证基本证书信息，包括：

① 证书是用来自证书 $i-1$ 的对象公钥签名的。

② 证书有效期包括时间 T。

③ 证书在时间 T 时未被撤销并且目前不处于时间 T 之前就开始的保留状态。

④ 对象和发布者名称链的正确性。

(2) 验证对象名称和 subjectAltName 扩展与限制子树状态变量是一致的。

(3) 验证对象名称和 subjectAltName 扩展与排除子树状态变量是一致的。

(4) 验证策略信息与可接受的策略集是一致的：

① 如果证书策略扩展标志为关键的，策略扩展和可接受策略的交集为非空。

② 可接受策略集用其新值对交集结果赋值。

(5) 验证可接受策略集和初始策略集的交集是非空的。

(6) 识别和处理证书中出现的任何其他的关键性扩展。

(7) 验证证书是一个 CA 证书。

(8) 如果证书中出现 permittedSubtress，则将限制子树状态变量设置为其之前的值与在扩展域中指出的值的交集。

(9) 如果 excludedSubtrees 出现在证书中，则将排除子树状态变量设置为其之前的值与在扩展中指出的值的并集。

(10) 如果一个策略约束扩展包含在证书中，显式策略和策略映射状态变量的修改包括：

① 如果 requireExplicitPolicy 出现且值为 r，那么显式策略状态变量被设置为其当值与 r 和 i 之和中的最小值。

② 如果 inhibitPolicyMapping 出现且其值为 q，那么策略映射状态变量被设置为其他

前值与 q 和 i 之和中的最小值。

(11) 如果密钥使用扩展是关键的，那么保证了 keyCertSign 比特位是设置的。

如果上述的任一项检查失败了，那么过程终止，返回一个错误指示和一个适当的理由。如果上述的所有项通过终端证书的检验，那么过程终止，返回一个成功指示和一个在证书集中遇到的所有策略的限定值集。

2. 扩展路径检验

上述提出的路径检验算法基于多个简化假设，当假设不成立时此算法可扩展。扩展办法是通过向检验模块提供自签名证书集。在这种情况下，一个有效路径可开始于任一个自签名证书。任一特定密钥的可信路径的限制可合并进自签名证书的扩展中。此时，自签名证书允许路径检验模块自动合并安全策略和要求。也可为上述证书路径处理过程说明一个扩展版本，在这个扩展版本中，过程的附加输入是多个策略证明授权(PCA)名称链和其在检验路径中的位置。在提名的 PCA 的位置上，CA 的名称依照此链进行比较。如果找到了一个可识别的 PCA 名称，那么 SubordinateToCA 的一个约束将隐含地假定证明路径的剩余部分和后续的处理。如果没有找到一个有效的 PCA 名称，并且如果证明路径不能为识别策略的基础所验证，那么证明路径被视为无效。

13.2.6 算法支持

RFC2459 中描述了 X.509 V3 所使用的 Hash 函数，用于证书和 CRL 签名的数字签名算法，以及为证书中公钥标识 OIDs 的算法。

RFC2459 包括了 3 个 Hash 函数，即 SHA-1、MD2 和 MD5。RFC2459 包括了 3 个基于 RSA 加密算法的数字签名算法，这些数字签名算法将 RSA 加密算法分别与 MD2、MD5 和 SHA-1 相结合，并在 RFC2313 中给出了具体定义。RFC2459 描述的证书可为任一公钥算法传递一个公钥，如 RSA、DSA 和 Diffie-Hellman 算法。

与 Kerberos 协议相比，X.509 协议有一个很大的优点，即 X.509 不需要物理上安全的在线认证服务器，因为一个证书包含了一个证书认证机构 CA 的签名。公钥证书可通过使用一个不可信的目录服务被离线地分配。

13.3 IPSec 协议

IP 层是 TCP/IP 网络中最关键的一层，IP 作为网络层协议，其安全机制可对其上层的各种应用服务提供透明的覆盖式安全保护。因此，IP 安全是整个 TCP/IP 安全的基础，是 Internet 网络安全的核心。

IP 协议 IPv4 由于最初设计时没有过多地考虑安全性，其中仅有的一个安全选项是为美国国防部(DoD)专用的，目前在 Internet 上使用的 IP 均未对安全选项进行处理。IPv4 缺乏对通信双方真实身份的验证能力，缺乏对网上传输的数据的完整性和机密性保护，并且由于 IP 地址可软件配置等灵活性以及基于源 IP 地址的认证机制，使得 IP 层存在着网络业务流易被监听和捕获、IP 地址欺骗、信息泄露和数据项被篡改等攻击，而 IP 是很难抵抗这些攻击的。为了实现安全 IP，Internet 工程任务组 IETF 于 1994 年开始了一项 IP 安全工程，专门成立了 IP 安全协议工作组 IPSEC，来制定和推动一套称为 IPSec(IP Security)的 IP 安

全协议标准。其目标就是把安全特征集成到 IP 层，以便对 Internet 的安全业务提供低层的支持。IETF 于 1995—2004 年相继公布了一系列关于 IPSec 的 RFC 建议标准。图 13.3 展示了 IPSec 协议在 TCP/IP 中的位置。

SMTP	HTTP	NNTP
TCP		
IP/IPSec		

图 13.3　IPSec 协议在 TCP/IP 中的位置

13.3.1　IPSec 体系结构

IPSec 是指 IETF 以 RFC 形式公布的一组安全 IP 协议集，是在 IP 包级为 IP 业务提供保护的安全协议标准，其基本目的就是把安全机制引入 IP 协议，通过使用现代密码学方法支持机密性和可认证性服务，使用户能有选择地使用，并得到所期望的安全服务。IPSec 将几种安全技术结合形成一个比较完整的安全体系结构，它通过在 IP 协议中增加两个基于密码的安全协议——认证头（AH）和封装安全载荷（ESP）来支持 IP 数据项的可认证性、完整性和机密性。通过 IP 安全协议和密钥管理协议构建起 IP 层安全体系结构的框架，能保护所有基于 IP 的服务或应用。并且当这些安全机制正确实现时，它不对用户、主机和其他未采用这些安全机制的 Internet 部件有负面影响。由于这些安全机制是独立于算法的，所以在选择和改变算法时不会影响其他部分的实现，对用户和上层应用程序是透明的。IPSec 的设计既适用于 IPv4 又适用于 IPv6，它在 IPv4 中作为一个建议的可选服务，对于 IPv6 是一项必须支持的功能。

1. 安全结构与功能

IPSec 的安全结构包括以下 4 个基本部分。

（1）安全协议——AH 和 ESP。

（2）安全关联（SA）。

（3）密钥交换——手工和自动（IKE）。

（4）认证和加密算法。

IPSec 由两大部分 3 类协议组成：IPSec 安全协议（AH/ESP）和密钥管理协议（IKE）。

IPSec 安全协议定义了如何通过在 IP 数据包中增加扩展头和字段来保证 IP 包的机密性、完整性和可认证性。包括两个安全协议：IP 认证头（IP AH）和 IP 封装安全载荷（IP ESP）。

一个密码算法的安全性寓于密钥之中。如果密钥能被破解或损害，那么攻击者就能读取所有的加密信息。因此，密钥的安全性是安全协议的一个非常重要的组成部分。此外，SA 的管理对于 IPSec 而言也是至关重要的。为便于 IPSec 协议的独立性，IPSec 将分开考虑 SA 和密钥的管理，这就是 Internet 密钥交换协议（IKE），IKE 定义了通信实体间进行身份认证、创建安全关联、协商加密算法以及生成共享会话密钥的方法。

此外，与 IPSec 密切相关的一个协议是 Internet 安全关联密钥管理协议（ISAKMP），它

为 Internet 环境下安全协议使用的安全关联和密钥的创建定义了一个标准通用框架，不仅仅适用于 IPSec。IKE 是 ISAKMP 关于 IPSec 的一个协议标准。

IPSec 可在主机或网关上实现，通过使系统能选择所需要的安全机制、决定使用的算法和密钥及使用的方式，在 IP 层提供所要求的安全服务。IPSec 能在主机之间、安全网关之间或主机与安全网关之间对一条或多条路径提供保护。IPSec 提供的安全功能或服务主要包括访问控制、无连接完整性、数据起源认证、抗重放攻击、机密性和有限的数据流机密性。

认证头(AH)协议支持访问控制、数据起源认证、无连接完整性和抗重放攻击服务；封装安全载荷(ESP)协议可以支持机密性、访问控制、有限的数据流机密性和抗重放攻击服务。可以同时使用 ESP 和 AH 以提供所有的安全服务。因为这些安全服务是在 IP 层提供的，所以能被任何高层协议如 TCP、UDP、ICMP、IGMP 等使用。

2. 安全关联与隧道

安全关联(SA)的概念是 IPSec 的基础。IPSec 使用的两种协议(AH 和 ESP)均使用 SA，IKE 协议(也就是 IPSec 使用的密钥管理协议)的一个主要功能就是 SA 的管理和维护。SA 是通信对等方之间对某些要素的一种协定，如 IPSec 协议、协议的操作模式(传输模式和隧道模式)、密码算法、密钥以及用于保护它们之间数据流的密钥的生存期。如果希望同时用 AH 和 ESP 来保护两对等方之间的数据流，则需要两个 SA：一个用于 AH，一个用于 ESP。定义用于 AH 或者 ESP 的隧道操作模式的 SA 称为隧道模式 SA，而定义用于传输操作模式的 SA 称为传输模式 SA。安全关联是单工的即单向的，因此，对于输出和输入的数据流就需要独立的 SA。术语 SA 束用于描述一组 SA，该组 SA 应用于始自或者到达特定主机的数据。

SA 是通过像 IKE 这样的密钥管理协议在通信对等方之间协商的。当一个 SA 的协商完成时，两个对等方都在它们的安全关联数据库(SAD)中存储有该 SA 参数。SA 的参数之一是它的生存期，它以一个时间间隔或者是 IPSec 协议利用该 SA 来处理的一定数量的字节数的形式存在。当一个 SA 的生存期过期，要么用一个新的 SA 来替换该 SA，要么终止该 SA。当一个 SA 终止时，它的条目将从 SAD 中删除。

SA 由一个三元组唯一地标识，该三元组包含一个安全参数索引(SPI)，一个用于输出处理 SA 的目的 IP 地址或者一个用于输入处理 SA 的源 IP 地址，以及一个特定的协议(如 AH 或者 ESP)。SPI 是用来作为唯一标识一个 SA 而生成的一个 32 位整数。它在 AH 和 ESP 头中传输。因此，IPSec 数据报的接收方可以容易地识别 SPI 并利用它连同源或者目的 IP 地址和协议来搜索 SAD，以确定与该数据报相关联的 SA 或者 SA 束。

隧道就是把一个包封装在另一个新包里面，整个源数据包作为新包的载荷部分，并在前面添加一个新的 IP 头。这个外部头的目的地址通常是 IPSec 防火墙、安全网关或路由器。通过隧道技术可以对外隐藏内部数据和网络细节。对 IPSec 而言，IP 隧道的直接目标就是对整个 IP 数据包提供完全的保护。IP 隧道如图 13.4 所示。

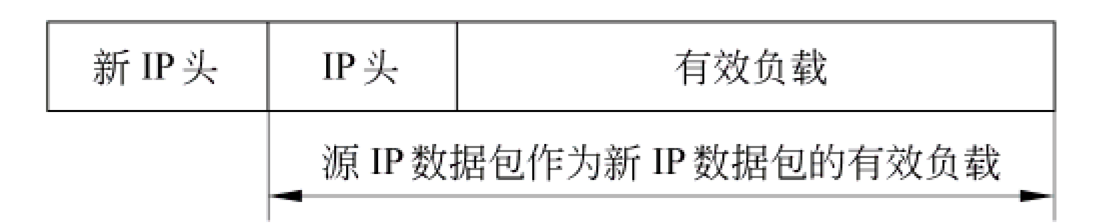


图 13.4　IP 隧道

此外，还有一个需要说明的概念就是“变换”。在 IPSec 中，一个变换是指一种安全机制的特定实现，如 ESP 的 DES-CBC 实现称为“DES-CBC ESP 变换”。

3. 安全关联数据库

为了处理 IPSec 数据流，IPSec 体系中定义了两个数据库：安全策略数据库(SPD)和安全关联数据库(SAD)。SPD 指定了用于到达或者源自特定主机或者网络的数据流的策略。SAD 中包含有与当前活动的 SA 相关的参数。对于 SPD 和 SAD，都需要单独的输入和输出数据库。

IPSec 协议要求在所有的通信流处理的过程中都必须查询 SPD，不管通信流是输入还是输出。SPD 中包含有一个策略条目的有序列表。通过使用一个或者多个选择符来确定每一个条目。IPSec 当前允许的选择符如下。

(1) 目的 IP 地址：目的 IP 地址可以是一个 32 位的 IPv4 或者是一个 128 位的 IPv6 地址。该地址可以是一个主机 IP 地址、广播地址、单播地址、任意播地址、多播组地址、地址范围、地址加子网掩码或者是通配地址。目的 IP 地址从 AH 和 ESP 或者 IP 头(如果没有对数据包应用 IPSec 的话)的目的 IP 地址域中得到。

(2) 源 IP 地址：同目的 IP 地址一样，源 IP 地址可以是一个 32 位的 IPv4 或者是一个 128 位的 IPv6 地址。同样，该地址可以是一个主机 IP 地址、广播地址、单播地址、任意播地址、多播组地址、地址范围、地址加子网掩码或者是通配地址。源 IP 地址从 AH 和 ESP 或者 IP 头的源 IP 地址域中得到。

(3) 传输层协议：传输层协议可以从 IPv4 协议或者 IPv6 协议的下一个头域中得到。

(4) 系统名：系统名可以是完整的 DNS 名，或者 E-mail 地址，如 bert. cs. mcgill. ca，或者 X. 500 DN，例如，cn=bert，ou=Computer Science，o=McGill University，st=Quebec，c=Canada。

(5) 用户 ID：用户 ID 可以是像 foo@cs. mcgill. ca 这样的完整的 DNS 或者是 X. 500 DN。

SPD 中的每一个条目包含有一个或者多个选择符和一个标志，该标志用于表明与条目中的选择符匹配的数据报是否丢弃、是否进行 IPSec 处理。如果应该对数据包进行 IPSec 处理，则条目中必须包含有一个指向 SA 内容的指针，其中详细说明了应用于匹配该策略条目的数据包的 IPSec 协议、操作模式及密码算法。

选择符与数据通信流相匹配的第一个条目将被应用到该通信中。如果没有发现匹配的条目，考虑中的通信数据包将被丢弃。因此，SPD 中的条目应该按照应用程序所希望的优先关系来排序。

SPD 中的条目决定了处理通信流的粒度。例如，策略可能规定与某特定的 SA 或 SA 束对应的 IPSec 服务，应该应用到任何源自任何源或者去往任何目的地的所有通信流，策略也可能规定应该基于特定的选择符来决定不同 SA 或者 SA 束的应用。SPD 在控制通过一个 IPSec 系统的所有通信中发挥着非常重要的作用。

SAD 中包含有现行的 SA 条目。每个 SA 由包含一个 SPI、一个源或者目的 IP 地址和一个 IPSec 协议的三元组来索引。此外，一个 SAD 条目包含有下面的域。

(1) 序列号计数器：这是一个 32 位整数，用于生成 AH 或者 ESP 头中的序列号域。

(2) 序列号溢出：这是一个标志，标识是否对序列号计数器的溢出进行审计，以及对于

特定的 SA，是否阻塞额外通信流的传输。

（3）抗重放窗口：使用一个 32 位计数器和一个位图来确定一个输入的 AH 或者 ESP 数据包是否是一个重放包。

（4）AH 认证密码算法和所需要的密钥。

（5）ESP 认证密码算法和所需要的密钥。

（6）ESP 加密算法、密钥、初始化向量（IV）和工作模式。

（7）IPSec 协议操作模式：该域表明将对 AH 和 ESP 通信应用何种 IPSec 协议操作模式（传输模式、隧道模式还是通配模式）。

（8）路径最大传输单元（PMTU）：PMTU 是可测量和可变化的。PMTU 是 IP 数据报经过一个特定的从源主机到目的主机的网络路径而无需分段的 IP 数据包的最大值。

（9）SA 的生存期：该域中包含有一个时间间隔，外加一个当它过期时该 SA 是被替代还是终止的标志。在该时间间隔内，某个 SA 必须用一个新的 SA 来替代或者终止。SA 的生存期有两种参数形式：一个是时间间隔形式，另一个是表示已用 IPSec 协议处理的字节数的字节计数形式。如果这两种参数都使用了，则以先过期的为准，即最先过期的参数优先。

对于输入和输出 IPSec 处理要保存单独的 SAD。对于输入或者输出通信，将搜索各自的 SAD 来查找与从数据包头域中解析出来的选择符相匹配的 SPI、源或者目的地址及 IPSec 协议。如果找到一个匹配的条目，则将该 SA 的参数与 AH 或 ESP 头中的适当的域相比较。如果头域与数据库中的 SA 参数相一致，就处理该数据包。然而，如果有任何差别，就丢弃该数据包。如果没有 SA 条目与选择符相匹配，并且如果数据包是一个输入包，就将它丢弃；然而，如果数据包是输出的，就将创建一个新的 SA 或者 SA 束，并其将存入输出 SAD 中。

SA 的生存期有两种类型的限制：软限制和硬限制。当达到一个软限制时，通信对等双方必须重新协商一个新的 SA 来代替已有的 SA。然而，已有的 SA 并不从数据库中删除，直到硬限制过期。与 SPD 不同，SAD 中的条目是无序的。然而，就像 SPD 中的查找一样，在 SAD 中找到的第一个匹配条目将被应用于与特定的 SA 关联的数据包的 IPSec 处理。

13.3.2 认证头协议

IP 协议本身缺乏安全性。用来提供 IP 数据报完整性的认证机制是非常初级的，仅用 IP 头中的校验和域来保证 IP 数据报的完整性。设计认证头（AH）协议的目的是用来增加 IP 数据报的安全性。AH 协议提供无连接的完整性、数据起源认证和抗重放保护服务。然而，AH 不提供任何机密性服务，它不加密所保护的数据包。AH 的作用是为 IP 数据流提供高强度的密码认证，以确保被修改过的数据包可以被检查出来。AH 使用消息认证码（MAC）对 IP 进行认证，最常用的 MAC 是 HMAC。

因为生成 IP 数据报的消息摘要需要密钥，所以 IPSec 的通信双方需要有共享密钥。这里假设：如果采用的密钥不同，对一个 MAC 输入指定数据计算出相同的消息摘要是计算上不可行的。于是，只有共享密钥的通信双方才可以采用预先定义的 MAC 对一个确定的消息生成确定的认证数据。

1. AH 格式

AH 由 5 个固定长度域和 1 个变长的认证数据域组成。图 13.5 说明了这些域在一个 AH 中的相对位置。

下一个头	载荷长度	保留
安全参数索引(SPI)		
序列号域		
认证数据(变长)		

图 13.5　AH 格式

下面是这些域的说明。

(1) 下一个头(Next Header)：这个 8b 的域指出 AH 后的下一个载荷的类型。例如，如果 AH 后面是一个 ESP 载荷，这个域将包含值 50。如果在所说的 AH 后面是另一个 AH，那这个域将包含值 51。RFC1700 中包含了已分配的 IP 协议值信息。

(2) 载荷长度(Payload length)：这个 8b 的域包含以 32b 为单位的 AH 的长度减 2。为什么要减 2 呢？AH 实际上是一个 IPv6 扩展头，IPv6 规范 RFC1883 中规定计算扩展头长度时应首先从头长度中减去一个 64b 的字。由于载荷长度用 32b 度量，两个 32b 字也就相当于一个 64b 字，因此要从总认证头长度中减去 2。

(3) 保留(Reserved)：这个 16b 的域被保留供将来使用。AH 规范 RFC2402 中规定这个域被置为 0。

(4) 安全参数索引(SPI)：SPI 是一个 32b 的整数，用于和源地址或目的地址及 IPSec 协议(AH 或 ESP)共同唯一标识一个数据报所属的数据流的安全关联(SA)。SA 是通信双方达成的一个协定，它规定了采用的 IPSec 协议、协议操作模式、密码算法、密钥以及用来保护它们之间通信的密钥的生存期。关于 SPI 域的整数值，1～255 被 IANA(Internet Assigned Number Authority)留作将来使用；0 被保留用于本地和具体实现使用。所以说目前有效的 SPI 值是 $256 \sim 2^{32}-1$。

(5) 序列号域(Sequence Number Field)：这个域包含有一个作为单调增加的计数器的 32 位无符号整数。当 SA 建立时，发送者和接收者的序列号值被初始化为 0。通信双方每使用一个特定的 SA 发出 1 个数据报就将它们相应的序列号加 1。序列号用来防止对数据包的重放，重放指的是数据报被攻击者截取并重新传送。AH 规范强制发送者总要发送序列号到接收者；而接收者可以选择不使用抗重放特性，这时它不理会进入的数据流中数据报的序列号。如果接收端主机启用抗重放功能，它使用滑动接收窗口机制检测重放包。具体的滑动窗口因不同的 IPSec 实现而不同；然而，一般来说滑动窗口具有以下功能。窗口长度最小为 32b。窗口的右边界代表一特定 SA 所接收到的验证有效的最大序列号。序列号小于窗口左边界的包将被丢弃。将序列号值位于窗口之内的数据包将被与位于窗口内的接收到的数据包相比较。如果接收到的数据包的序列号位于窗口内并且数据包是新的，或者它的序列号大于窗口右边界且小于 2^{32}，那么接收主机继续处理计算认证数据。对于一个特定的 SA，它的序列号不能循环；所以在一个特定的 SA 传输的数据包的数目达到 2^{32} 之前，必须协商一个新的 SA 及新的密钥。

(6) 认证数据：这个变长域包含数据报的认证数据，该认证数据被称为完整性校验值(ICV)。对于IPv4数据报，这个域的长度必须是32的整数倍；对于IPv6数据报，这个域的长度必须是64的整数倍。用来生成ICV的算法由SA指定。用来计算ICV的可用的算法因IPSec的实现的不同而不同；然而为了保证互操作性，AH强制所有的IPSec实现必须包含两个MAC：HMAC-MD5和HMAC-SHA-1。如果一个IPv4数据报的ICV域的长度不是32的整数倍，或一个IPv6数据报的ICV域的长度不是64的整数倍，必须添加填充比特使ICV域的长度达到所需要的长度。

2. AH的操作模式

AH头的位置依赖于AH的操作模式。AH有两种操作模式，即传输模式和隧道模式。使用隧道模式时，AH可以以一种嵌套的方式来应用，也可以和ESP组合使用或单独应用。

在传输模式中，AH被插在IP头之后但在所有的传输层协议之前，或所有其他IPSec协议头之前。因此，在传输模式中，对IPv4而言，AH被插在变长可选域之后。图13.6说明了在传输模式中AH相对于其他IPv4头部域的位置。

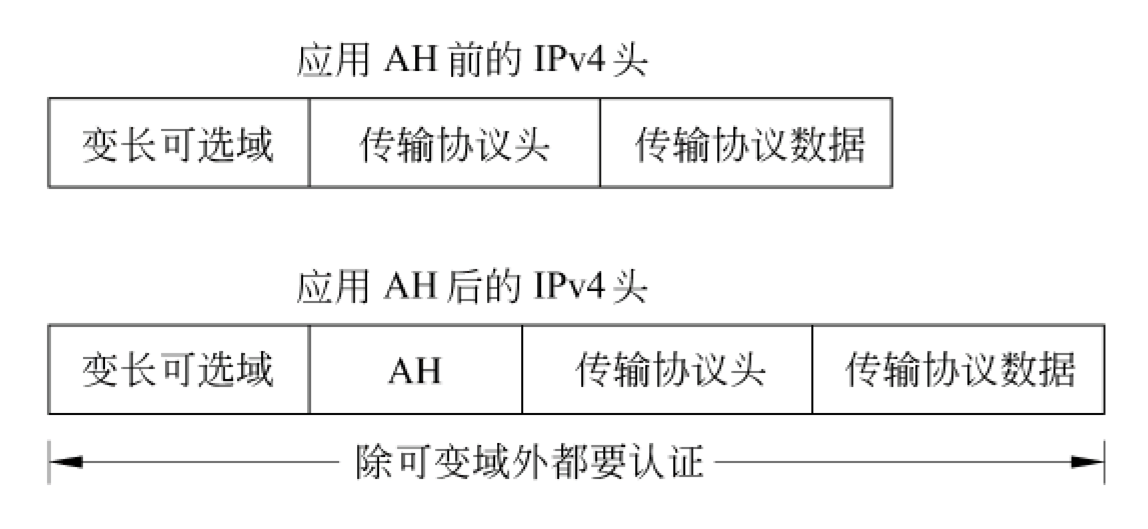


图13.6 AH在传输模式中相对于其他IPv4头部域的位置

在IPv6的传输模式中，AH被插在逐跳、路由和分段扩展头的后面；目的选项扩展头可以放在AH头的前面或后面。如果目的选项头将被出现在IPv6目的地址域的第一个目的主机以及其后的路由头中列出的目的主机列表处理，它应该紧接在逐跳头之后处理，因此最终要放在AH之前。可是，如果它仅被目的主机处理，它应该放在AH之后。图13.7说明了在传输模式中，AH头相对于其他IPv6扩展头的位置。

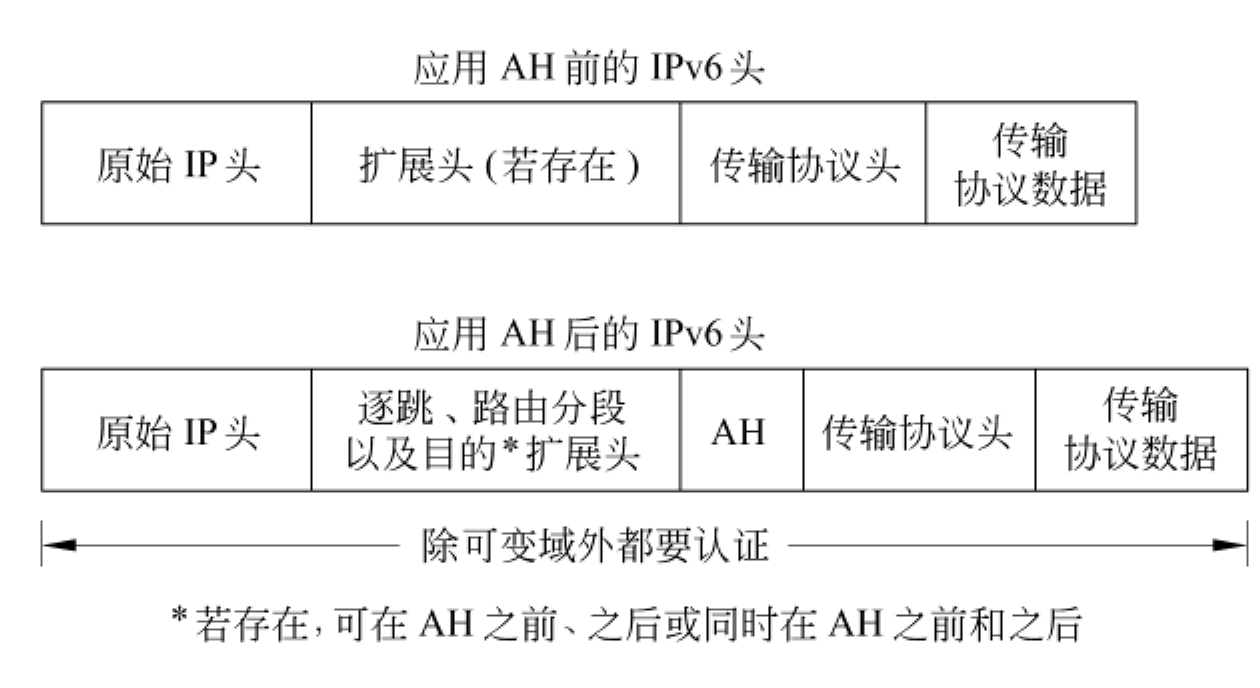
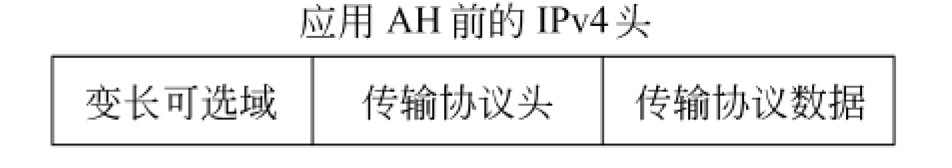


图13.7 传输模式下AH相对于其他IPv6扩展头的位置

在隧道模式中，AH插在原始的IP头之前，另外生成一个新的IP头放在AH之前。在图13.8中针对IPv4数据报对这一点进行了图解。在IPv6数据报中，除了新的IP头外，原

来数据报的扩展头也被插在 AH 前面。图 13.9 中说明了 IPv6 数据报的这些信息。

应用 AH 前的 IPv4 头



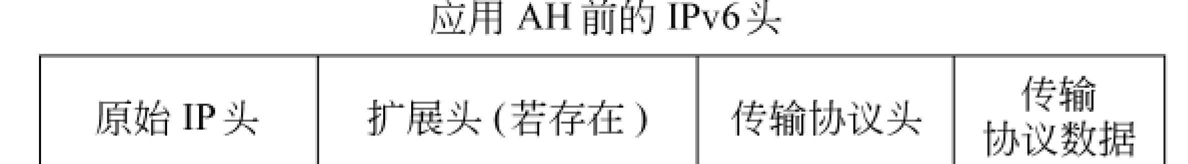
应用 AH 后的 IPv4 头

新 IP 头的可选域	AH	原始 IP 头的可选域	传输协议头	传输协议数据

除了新 IP 头中的可变域都要认证

图 13.8　隧道模式下 AH 相对于其他 IPv4 头的位置

应用 AH 前的 IPv6 头



应用 AH 后的 IPv6 头

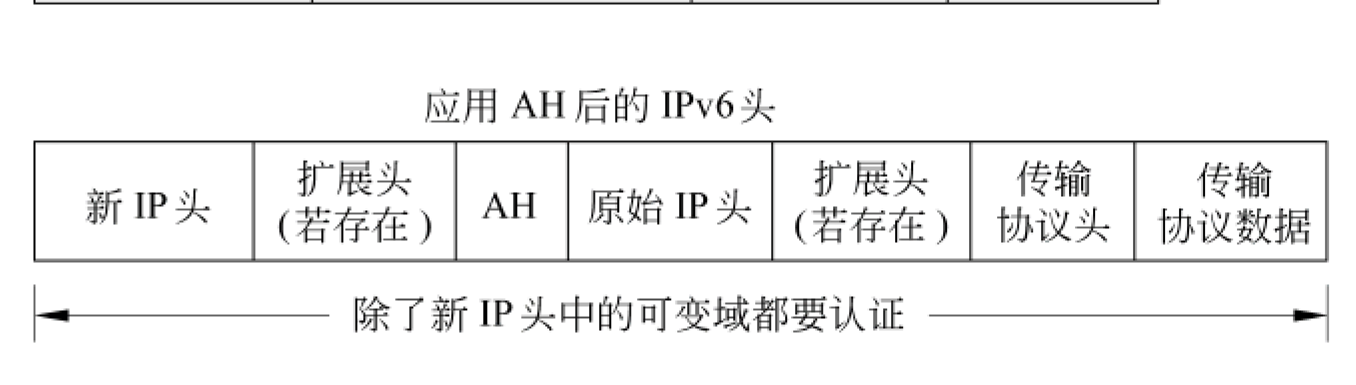


图 13.9　隧道模式下 AH 相对于其他 IPv6 扩展头的位置

3. 完整性校验值的计算

完整性校验值(ICV)是 AH 或 ESP 用来验证 IP 数据报的完整性所用的验证数据。数据包的 ICV 是用 MAC 生成的。在 MAC 的生成中,密钥是必需的。IPSec 的通信各方必须能够产生 ICV 来校验交互的数据报的完整性,因此,通信双方需要共享的密钥。在 IPSec 通信双方通信之前,它们需要协商一个 SA。SA 是通信双方关于参数选择的一个协议,这些参数包括用来认证或加密(对 ESP 而言)数据流的密码算法、指定密码算法使用的密钥、IPSec 协议(AH 或 ESP)、协议的操作模式及 SA 的生命期。

当通信双方建立了一个 SA 后,它们就有了所有用来计算它们交换的数据报的 ICV 值的参数。ICV 的计算涉及整个 IP 头,然而有些域在从源到目的的传输过程中可能会改变,所以在计算 ICV 时将这些域设为 0。下面分别说明 IPv4 和 IPv6 数据包头的可变域和不变域。

IPv4 头的可变域如下。

(1) 服务类型(Type of Service):8b 的服务类型(TOS)域表明了数据报所属数据流的需求,这个需求是针对延迟、吞吐量和可靠性的组合而言的。IP 协议并不认为这个域是可变的,但 IPSec 将它作为可变域对待,因为现有的一些路由器会改变这些域。

(2) 标志(FLAGS):IP 头中有 3 个标志位。第一个是 DF(不可分段)位,当这个位被设置时表明不允许对数据报分段。第二个是 MF(更多分段)位,当这个位为 0 时表明数据包的最后一个数据分段到达。第三个被留作将来使用。ICV 的计算不包括这个域,因为即使数据源没有设置这个位,中间的路由器也可能设置 DF 位。

(3) 分段偏移量(Fragment Offset):这个13b的域表明一个分段在它所属的数据报中的位置。AH仅应用于非分段的IP包,所以在计算ICV之前,必须将这个域设为0。

(4) TTL:这个8b的域被用来限制一个数据报的生命期,因此可以防止数据报在一个网段中无限循环。数据报在经过每个路由器时,它的TTL值都被路由器减小。目的主机无法预计TTL值,所以这个域被排除在ICV的计算之外。

(5) 头校验值(Header Checksum):这个16b的域保存IP头的校验和。如果IP头的任一个域发生了变化,校验和的值也会变化,所以在计算ICV之前,必须将这个域设为0。

(6) 可选项(Options):这个变长域存放数据包的可选信息,比如安全性和处理限制。这个域很少被使用,并且被大多数的路由器忽略。所以大部分的IPSec实现在计算ICV时都不包含可选项。

IPv4头的不变域如下。

(1) 版本(Version)。

(2) 头长度(IHL)。

(3) 总长度(Total Length)。

(4) 标识(Identification)。

(5) 协议(Protocol)。

(6) 源地址(Source Address)。

(7) 目的地址(Destination Address)。

(8) 数据(Data)(被封装的传输协议头和数据)。

IPv6头的可变域如下。

(1) 优先级(Priority):这个4b的域表明一个数据包要求的服务质量。这个域可能会被中间的路由器改变,所以它没有被包含在ICV的计算中。

(2) 流标签(Flow label):这个24b的域是一个实验性的域。大多数应用忽略它,最终IPSec强制在计算ICV值之前将其设为0。

(3) 跳数限制(Hop limit):这个跳数限制域和IPv4的TTL域相同。它在经过每个路由器时都被减1。

逐跳和目的选项扩展头中包含1b,用来表明这个选项在传输过程中是否可能发生变化。可以通过设置和取消这些比特来指定这些扩展头可变还是不可变。逐跳扩展头用来携带可选的路由信息,数据报从源到目的沿途的每个节点都必须检查这些信息。目的选项扩展头携带仅需由目的节点检查的可选信息。

IPv6头的不变域如下。

(1) 版本(Version)。

(2) 载荷长度(Payload Length)。

(3) 下一个头(Next Header)。

(4) 源地址(Source Address)。

(5) 目的地址(Destination Address)。

路由扩展头有可变但可预测的特性。IPv6数据源用这个扩展头列出数据包从源到目的途径中的一个或多个网关。路由扩展头中的地址域可能在传输中被重排;然而数据包在目的主机时的内容的形式对发送者和中间节点而言却是已知的。所以,可以由发送者排列

地址域使其和发送到目的地时相同。因此是在计算 ICV 时包含了这个扩展头。

如果有分段扩展头的话，它对 IPSec 而言是不可见的，因为它在外出的 IPSec 处理后才出现；而重组工作在进入 IPSec 处理之前进行。所以，在计算 ICV 时这个扩展头不予考虑。

把 IP 的可变域和 IPv6 可变扩展头设为 0 后，整个 IP 数据报以一个比特串的形式作为 MAC 的输入。MAC 采用指定的密钥生成 ICV。值得注意的是，在隧道模式中，内部 IP 头的可变域未被设为 0，因为只有外部 IP 头的可变域才可能在传输过程中被修改。

ICV 的长度依赖于使用的 MAC 算法。例如，对于 HMAC-MD5，ICV 是 128b；而对于 HMAC-SHA1，ICV 是 160b。

如果一个 MAC 算法生成的 IPv4 数据报的 ICV 长度不是 32 的整数倍，或生成的 IPv6 数据报的 ICV 长度不是 64 的整数倍，则需要增加填充比特，使 ICV 达到 32（IPv4）或 64（IPv6）的整数倍。

计算出 ICV（有必要的话还得对它进行填充）之后，把它放在认证数据域中，然后数据报将被发送到目的地址。

4. AH 的处理

虽然处理 AH 数据流的一些具体细节可能与 IPSec 的具体实现有关，但对 AH 的规范而言，每个实现都应该遵循一些共同的东西。

(1) 外出处理

当一个 IPSec 实现从 IP 协议栈中收到外出的数据包，它使用相应的选择符（目的 IP 地址、端口和传输协议等）来查找安全策略库（SPD），并确认对数据流应用怎样的策略。如果需要对数据包进行 IPSec 处理，并且到目的主机的一个 SA 或 SA 束已经建立，那么符合数据包选择符的 SPD 将指向外出 SA 数据库的一个相应的 SA 束。如果 SA 还未建立，IPSec 的实现将调用 IKE 协商一个 SA，并将其连接到 SPD 条目上，然后 SA 用于以下数据包的处理。

① 产生或增加序列号值：序列号用来防止以前发送过的包的重放。当一个新的 SA 建立时，发送者将序列号计数器初始化为 0。发送者每发送一个包，它就将序列号加 1 并将结果插入 AH 头中的序列号域。

② 如前所述计算 ICV。

③ 转发数据包到目的地。

(2) 进入处理

当一个设置了 MF 位的数据报到达一个 IPSec 目的节点时，这表明其他的分段还没有到达。IPSec 应用等待直到一个有着相同序列号但 MF 位未设置的分段到达。IP 分段重组之后要进行以下步骤的操作。

① 使用 IP 头（如果在隧道模式下是外层 IP 头）中的 SPI、目的 IP 地址及 IPSec 协议，在进入的 SA 数据库中查找数据包所属数据流的 SA。如果查找失败，它将抛弃该数据包并记录事件。记录日志的总信息量通常取决于 IPSec 的具体实现。然而，至少应该记录抛弃数据包的时间、源节点的 IP 地址和抛弃的原因。

② 使用第①步中查找到的 SA 进行 IPSec 处理。首先要确定 IP 头（隧道模式中是内部头）中的选择符和 SA 中的选择符是否匹配。如果选择符不匹配，应用将抛弃数据包并审计事件。如果选择符匹配，IPSec 应用跟踪 SA 以及它相对于其他的 SA 应用的顺序，并重复

①、②步直到遇到传输层协议，或一个非 IPSec 扩展头(对 IPv6 而言)。

③ 使用数据包中的选择符在进入 SPD 中查找一条与选择符匹配的策略。

④ 检查步骤①、②中查到的 SA 是否和步骤③中查找到的策略匹配。如果匹配失败，重复④、⑤步直到处理完所有的策略条目，或直到匹配成功。

⑤ 如果启用了抗重放的功能，如前面章节所说的那样，使用 SA 的抗重放窗口检查数据包是否是重放包。如果是重放包，则抛弃它并审计事件。

⑥ 使用 SA 指定的 MAC 算法计算数据包的 ICV，并将它和认证数据域中的值相比较。如果两个值不同，则抛弃数据包并审计事件。

在这些步骤之后，如果数据包未被抛弃，它将被发送到 IP 协议栈的传输层或转发到指定的节点。

13.3.3 封装安全载荷协议

与认证头(AH)协议一样，设计封装安全载荷(ESP)协议的目的也是为了提高 Internet 协议(IP)的安全性。ESP 提供数据机密性、数据源认证、无连接完整性、抗重放攻击和有限的数据流机密性等服务。实际上，ESP 提供的服务和 AH 的类似，但增加了两个额外的服务：数据机密性服务和有限的数据流机密性服务。数据机密性服务通过使用密码算法加密 IP 数据报的相关部分来实现。数据流机密性服务由隧道模式下的机密性服务提供。

ESP 中用来加密数据报的密码算法都毫无例外地使用对称密码算法。公钥密码算法涉及计算量非常大的大整数模指数运算，大整数的规模超过 300 位十进制数字。而对称密码算法主要使用初级的操作(异或、逐位与、位循环等)，无论以软件还是硬件方式执行都非常有效。所以相对公钥密码算法而言，对称密码算法的加、解密吞吐量要大得多。

ESP 通过使用消息认证码(MAC)提供认证服务。对于加密和认证算法的选择，因 IPSec 的实现不同而不同；然而为了保证互操作性，ESP 规范 RFC2406 中规定了每个 IPSec 实现要强制实现的算法。在撰写规范时，强制实现的加密算法是 DES 的 CBC 模式和 null 加密算法，而认证算法是 HMAC-MD5，HMAC-SHA-1 和 null 认证算法。null 加密和 null 认证算法分别是不加密和不认证选项。null 算法选项是强制实现的，因为 ESP 加密和认证服务是可选的。然而，值得注意的是，null 加密和 null 认证算法不可以同时使用。换句话讲，如果采用了 ESP，或者要用到加密，或者要用到认证，或者同时使用。DES 是个弱加密算法，很少用于 VPN 解决方案。人们希望可以在修订 RFC2406 时用 AES 代替 DES 作为强制实现的算法。其他常用的算法是 CAST-128 和 IDEA。

ESP 可以单独应用或以嵌套的方式使用，也可以和 AH 结合使用。

1. ESP 数据包格式

ESP 数据包由 4 个固定长度的域和 3 个变长域组成。这个协议的包格式如图 13.10 所示。有关域的描述如下。

(1) 安全参数索引(SPI)：SPI 是一个 32b 的整数，它同源地址、目的地址和 IPSec 协议(ESP 和 AH)结合起来唯一标识数据报所属的数据流的安全关联(SA)。SA 是通信双方关于一些实体的一个协议，如用来提供 ESP 机密性服务的加密算法、认证算法、密钥、IPSec 协议操作模式和 SA 的生存期。关于 SPI 域的整数值，1～255 被 IANA 留作将来使用；0 被保

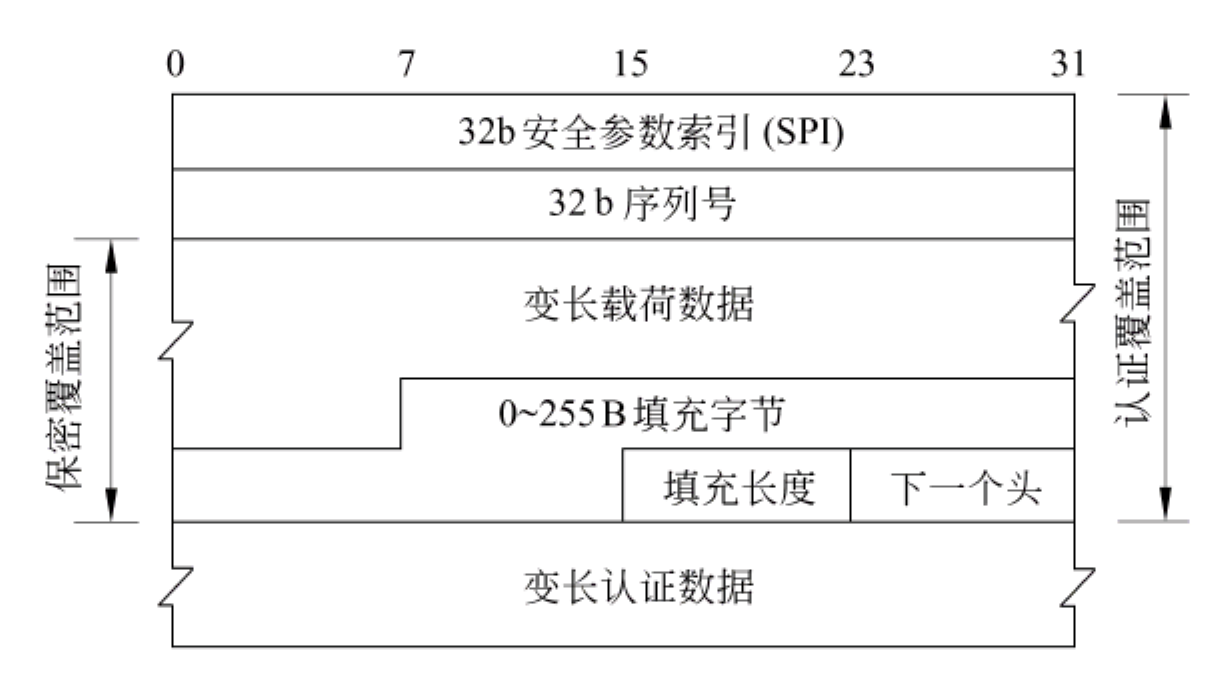


图 13.10　ESP 的数据包格式

留用于本地和具体实现使用。所以说目前有效的 SPI 值是 256～($2^{32}-1$)。这个域和 AH 的 SPI 类似。

(2) 序列号域(Sequence Number Field)：与 AH 的情况类似，这个域包含着一个作为单调增加的计数器的 32b 无符号整数。当 SA 建立时，发送者和接收者的序列号值被初始化为 0。对一个特定的 SA，双方每发出一个数据包就将它们的序列号增加 1。序列号用来防止数据报的重放。对一个特定的 SA，它的序列号不能循环，所以，在一个特定的 SA 传输的数据包的数量达到 2^{32}之前，必须协商一个新的 SA 及新的密钥。规范强制发送者总要发送序列号到接收者；而接收者可以选择不使用抗重放特性，这时它不理会进入的数据流中数据报的序列号。如果接收端主机启用抗重放功能，它使用滑动接收窗口机制检测重放包。具体的滑动窗口因不同的 IPSec 实现而不同；然而一般来说，滑动窗口具有以下所述的功能。窗口最小为 32b。窗口的右边界代表一特定 SA 所接收到的经验证有效的最大序列号。将序列号值位于窗口之内的数据包与位于窗口内的接收到的数据包相比较。如果接收到的数据包的序列号位于窗口内并且数据包是新的，或者它的序列号大于窗口右边界且小于 2^{32}，那么接收主机继续处理计算认证数据。不然的话将抛弃数据包并审计事件。

(3) 载荷数据(Payload Data)：这是一个变长域。如果使用机密性服务的话，其中就包含有实际的载荷数据(就是说，数据报加密部分的密文)。这个域是必须有的，不管涉及的 SA 是否需要机密性服务。如果采用的加密算法需要初始化向量(IV)，它将在载荷域中传输，并且算法的规范要指明 IV 的长度和它在载荷数据域中的位置。IV 用于某种分组密码的工作模式以确保前一部分相似的明文(如 IP 数据报头)生成不同的密文。载荷数据域的长度以 bit 为单位并且必须是 8 的整数倍。

(4) 填充(Padding)：如果有的话，这个域包含着填充比特，由加密算法使用或用于使填充长度域和 4B 字中的第 3 个字节对齐(见图 13.10)。这个域的长度是 0～255B。

(5) 填充长度(Pad length)：填充长度是一个 8b 的域，表明填充域中填充比特的长度。这个域的有效值是 0～255 间的整数。

(6) 下一个头(Next Header)：这个 8b 的域表明载荷中封装的数据类型。可能是一个 IPv6 扩展头或一个传输层协议。例如，值 6 表明载荷中封装的是 TCP 数据。IANA 是一个负责分配 IP 协议值的组织。IANA 的主页是 http://www.iana.org。

(7) 认证数据(Authentication Data)：这个变长域中存放 ICV，如图 13.10 所示，它是对除认证数据域外的 ESP 包计算出来的。这个域的实际长度取决于使用的认证算法，例

如，如果使用 HMAC-MD5，则认证数据域是 128b；如果使用 HMAC-SHA-1，则为 160b。认证数据域是可选的，仅当指定的 SA 要求 ESP 认证服务时才包含它。

2. ESP 的操作模式

与 AH 的情况一样，ESP 在数据包中的位置取决于 ESP 的操作模式。ESP 共有两种操作模式，即传输模式和隧道模式。

在传输模式下，ESP 被插在 IP 头和所有的选项之后，但在传输层协议之前，或者在已应用的任意 IPSec 协议之前。所以，在 IPv4 传输模式下，ESP 被插在变长选项域之后。图 13.11 显示了 ESP在传输模式中相对于其他头部域的位置。图中，ESP 的头部域由 SPI 和序列号域组成，而 ESP 尾部由填充域、填充长度域和下一个头域组成。图中标明了数据报被加密和认证的部分。如果需要机密性服务，SPI 和序列号域不被加密，因为接收节点需要这些域来标志用来处理数据报的 SA，另外如果启动了抗重放服务，它们还被用来检验重放数据包。类似地，如果有认证数据域，那它不被加密，因为如果某个 SA 需要 ESP 认证服务，目的主机在处理这个数据报之前首先用这个域来验证数据报的完整性。

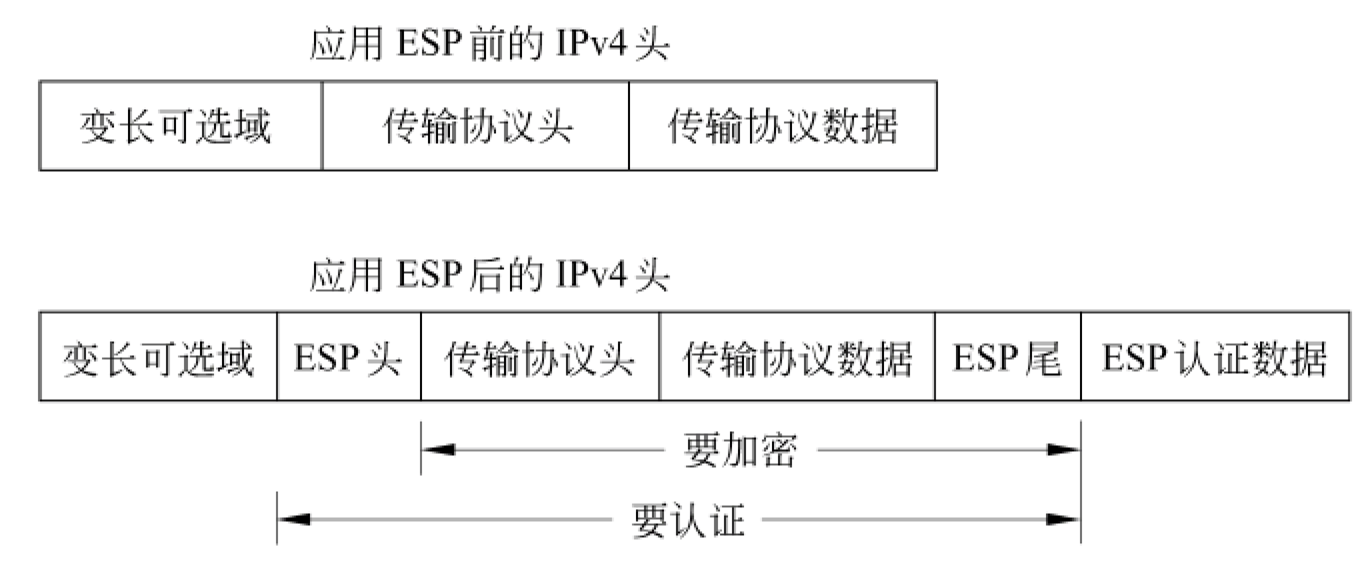


图 13.11 传输模式下 ESP 相对于其他 IPv4 头的位置

对于 IPv6 数据报，ESP 被插在逐跳、路由和分段扩展头之后，目的选项扩展头可以放在 ESP 头的前边或后边。如果目标选项头被 IPv6 目标地址域的第一个目的主机以及由路由头列出的相继的目的主机处理，那目标选项头将放在 ESP 之前。然而，如果它仅被目的节点处理，它可以放在 ESP 之后。图 13.12 说明了在传输操作模式下，ESP 相对于其他 IPv6 扩展头的位置。

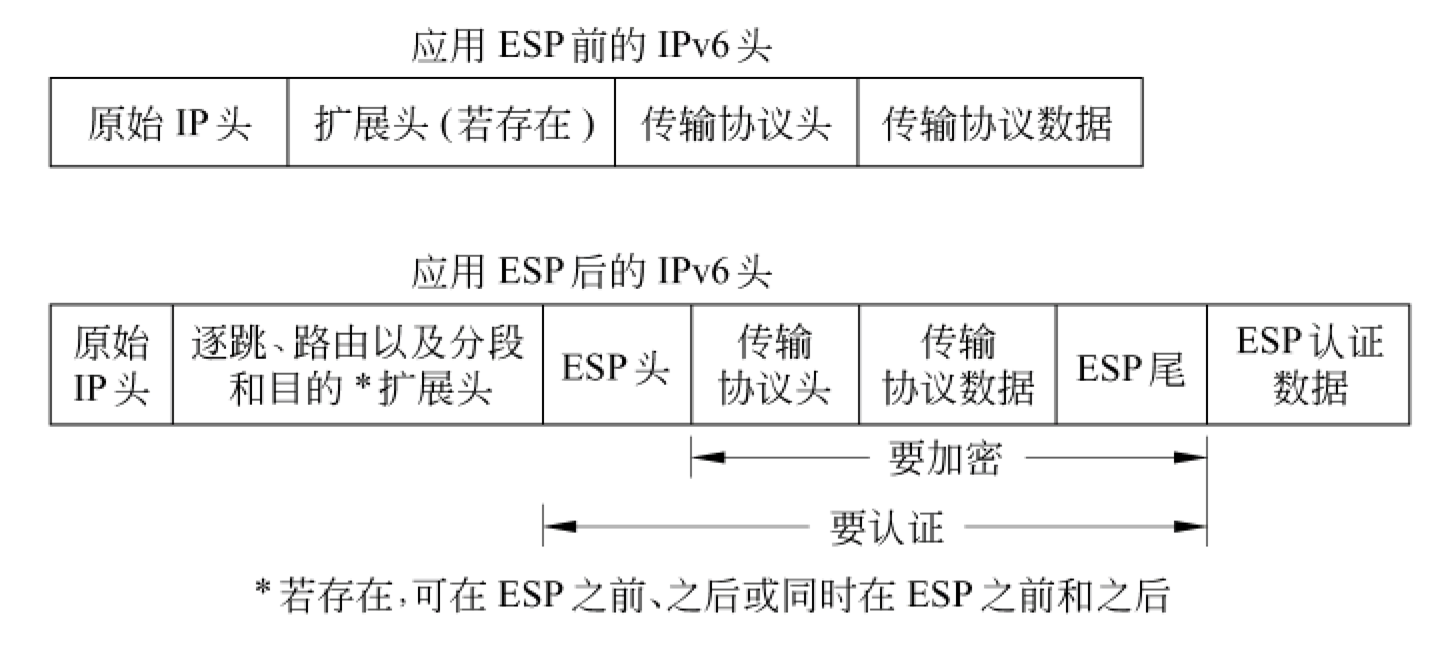


图 13.12 在传输模式下 ESP 相对于其他 IPv6 扩展头的位置

关于 ESP 认证服务有一点需要强调的是，与 AH 不同，ESP 不对整个 IP 数据报进行认

证。使用私有 IP 地址(通过 Internet)或位于安全网关之后的主机间的通信可被 ESP 认证服务保护,因为 IP 头中的源和目的以及其他的域未被认证。于是,NAT(Network Address Translation)和安全网关可以改变数据报相应的 IP 头的域。并且如果修改之后的头部校验和计算正确且 ESP 头部未被修改,目的节点将成功认证数据报。

然而,ESP 提供的这种灵活性导致了它的弱点。除了 ESP 头部外,在从源到目的的传输过程中 IP 头的任何域都可以被修改,如果修改后头部校验和计算正确(如果涉及的 SA 只使用 ESP 认证和加密服务),目的主机将无法检测到修改。这样,ESP 传输模式下认证服务所提供的安全性就不如 AH 传输模式。所以,需要更高安全级并且通信双方使用公开 IP 地址时,应该采用 AH 认证服务或者采用 AH 认证服务和 ESP 认证服务结合。

需要提到的一点是,传输模式下的 ESP 不提供数据流机密性服务,因为源和目的 IP 地址未被加密。

在隧道模式下,ESP 被插在原始 IP 头之前,并且生成一个新的 IP 头并将其插在 ESP 之前。在图 13.13 中对 IPv4 下的情况进行了说明。对 IPv6 数据报而言,除了新的 IP 头,原始 IPv6 数据报中的扩展头也被插在 ESP 头之前。图 13.14 说明了 IPv6 下的情况。内部 IP 头中包括真正的源地址(生成数据包的节点)和最终目的地址。外部的源及目的 IP 头域分别包含源及目的节点的安全网关。所以,内、外部 IP 头的源地址可能不同,目的地址也可能不同。

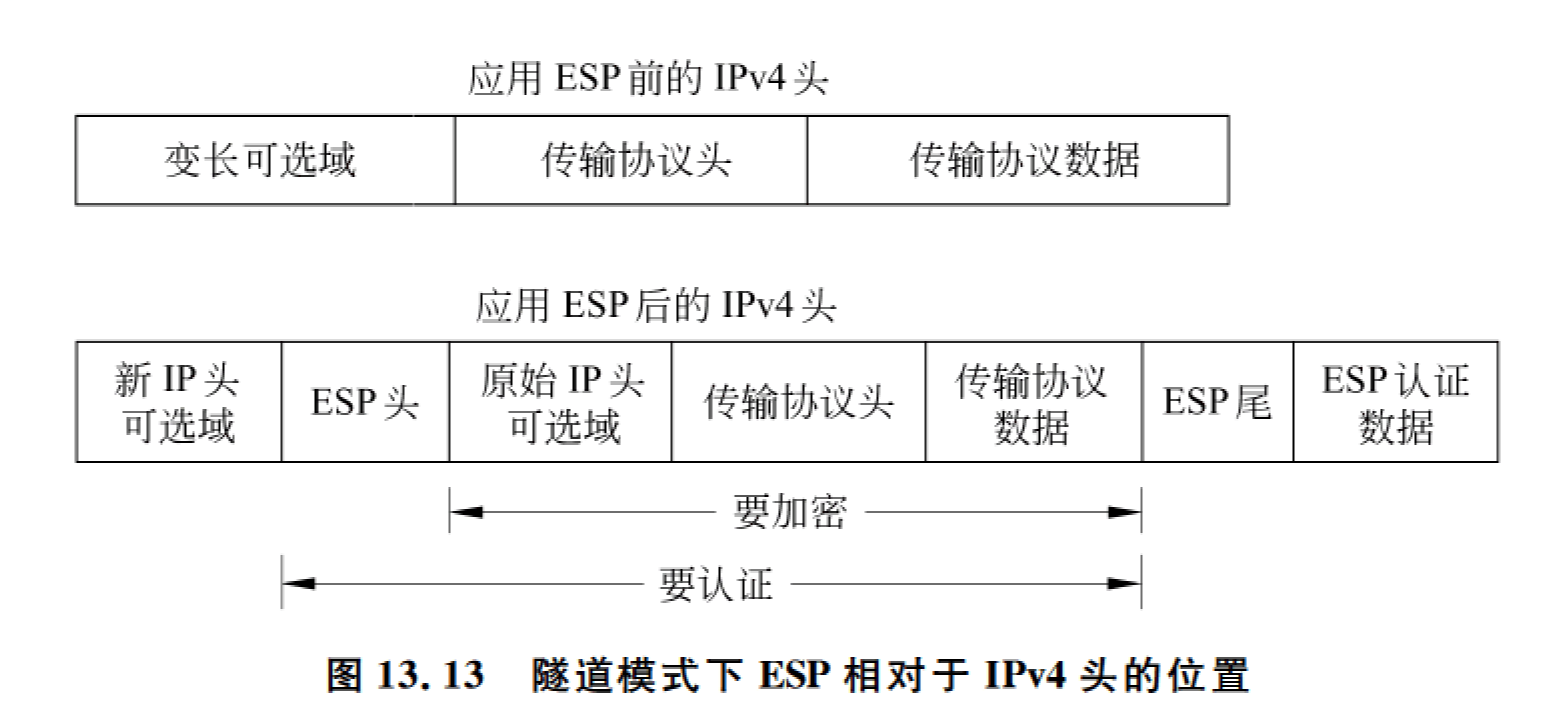


图 13.13　隧道模式下 ESP 相对于 IPv4 头的位置

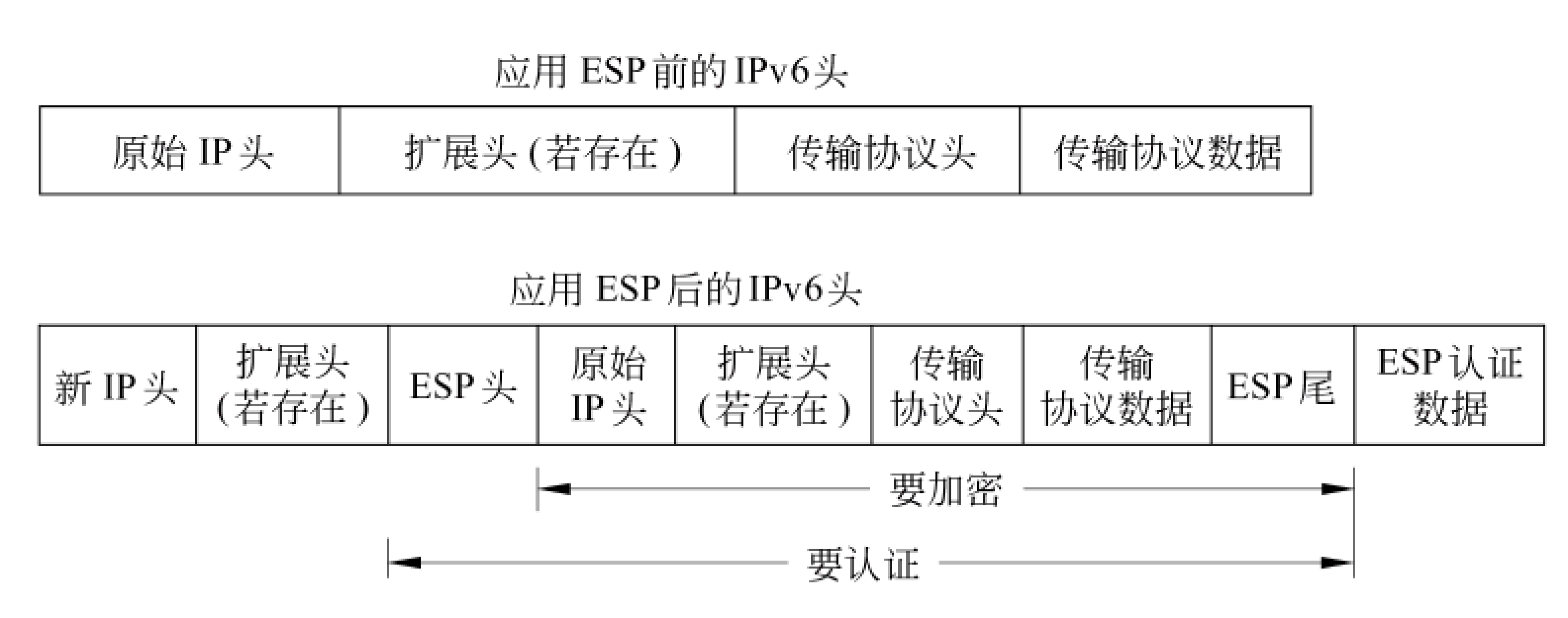


图 13.14　隧道模式下 ESP 相对于其他 IPv6 扩展头的位置

与传输模式的情况一样,ESP 头部域由 SPI 和序列号域组成,而 ESP 尾部域由填充域、填充长度域和下一个头域组成。IPv4 和 IPv6 数据报需要加密和认证部分在图 13.13 和图 13.14 中分别说明。ESP 头部和尾部域未被加密,因为这些域中包含一些信息,目的节点

需要利用这些信息查找用于处理数据报所属的数据流的 SA，或者在加密之前处理数据报时需要这些信息。

对于隧道模式的认证和机密性服务，要对整个内部 IP 头进行认证和加密。而外部 IP 头既未被认证也未被加密。

有一点非常重要，ESP 隧道模式认证和加密服务所提供的安全性要强于 ESP 传输模式；因为不像后者那样，前者认证和加密原始的 IP 头。而隧道模式服务将比传输模式服务占用更多的带宽，因为隧道模式在保护的数据报中插入了一个额外的 IP 头。所以如果带宽利用率非常重要的话，传输模式可能是更合适的选择。

尽管理论上 ESP 隧道模式认证提供的安全性不如 AH 传输模式或隧道模式的安全性(因为它不认证外部 IP 头)高，但它提供的安全性已经足够，因为用来处理数据包的信息是在内部 IP 头中。

同样值得注意的一点是，ESP 隧道模式的机密性服务，特别是在安全网关上实现时，它可以提供数据流机密性服务，因为包含 IP 数据包源地址的内部 IP 头被加密了。

3. ESP 的处理

处理 ESP 的过程如下。

(1) 外出处理

当一个 IPSec 实现接收到一个外出的数据包，它使用相应的选择符(目的 IP 地址和端口、传输协议等)查找安全策略库(SPD)并且确认哪些策略适用于数据流。如果需要 IPSec 处理并且一个 SA 或 SA 束已建立，与数据包选择符相匹配的 SPD 项将指向安全关联库(SAD)中的相应 SA。如果 SA 还未建立，IPSec 的实现将使用 IKE(Internet 密钥交换)协议协商一个 SA 并将其链接到 SPD 项。接下来 SA 将被用于进行下面的数据包的处理。

① 生成或增加序列号：序列号用于防止以前发送过的数据包的重放。当建立一个新的 SA 时，发送者将其序列号计数器初始化为 0。发送者每发送一个数据包，它将序列号加 1 并将序列号计数器的结果值插入 ESP 包的序列号域。

② 加密数据包：如果传输需要保密服务，SA 将指明使用的加密算法。可用的加密算法依赖于 IPSec 的实现，但是如前所述，目前仅使用了对称密码系统。因为相对于对称密码系统，公钥密码系统处理速度较慢。当加密算法需要初始化向量(IV)时，就像 DES 的 CBC 模式那样，初始化向量放在载荷数据域的前几个比特。当需要加密服务时，在计算完整性校验值(ICV)之前，必须先加密数据包。

③ 计算完整性校验值：如果数据包的 SA 指定应用 ESP 认证服务，ICV 的计算使用了除认证数据域外的所有 ESP 头域，而认证数据域用来存放 ICV。数据包的 SA 指定了用来生成 ICV 的消息认证码算法。可用的认证算法因不同的 IPSec 实现而不同。然而，为了互操作性 ESP 规范指定所有的实现必须支持 HMAC-MD5 和 HMAC-SHA-1。认证算法生成 ICV 时需要密钥。IKE 协议负责协商和建立必要的密钥和其他的 SA 参数。

④ 分段：如果需要分段，可以通过适当的方法取得从包的源到目的路径的最大传输单元(MTU)。然后数据包被分成适当的大小并发往目的节点。

(2) 进入处理

当一个数据包到达一个 IPSec 主机或安全网关时，如果更多分段(MF)比特被设置，就意味着还有其他的分段未到达。IPSec 应用等待直到收到一个数据包，它的序列号和前面

几个相同，但是 MF 比特未设置。之后它重组 IP 分段并执行以下步骤。

① 使用 IP 头中的目的 IP 地址和 IPSec 协议以及 ESP 头中的 SPI，在进入 SAD 中查找进入包的 SA。如果查找失败，抛弃该数据包并审计事件。

② 使用第①步中查找到的 SA 对 ESP 包进行处理。首先要检查已确定 IP 头中（在隧道模式下是内部头）的选择符是否和 SA 中的匹配。如果选择符不匹配，抛弃该数据包并审计事件。如果匹配，IPSec 应用跟踪 SA 以及它相对于其他 SA 的应用的顺序并重复步骤①、②，直到遇到一个传输层协议（对 IPv4 数据报而言）或一个非 IPSec 扩展头（对 IPv6 数据报而言）。

③ 使用包中的选择符在进入 SPD 中查找一条和包选择符匹配的策略。

④ 检查步骤①、②找到的 SA 是否和步骤③找到的策略匹配。如果匹配失败，则重复步骤④、⑤直到所有的策略被匹配完或匹配成功。

⑤ 如果启用了抗重放服务，使用本章前一部分讨论过的抗重放窗口来决定某个包是否是重放包。如果是重放包，抛弃该数据包并审计事件。

⑥ 如果 SA 指定需要认证服务，应用 SA 指定的认证算法和密钥生成数据包的 ICV，并和 ESP 认证数据域中的值相比较。如果两个值不同，抛弃该数据包并审计事件。

⑦ 如果 SA 指定需要加密服务，应用 SA 指定的加密算法和密钥解密数据包。一般而言，解密处理对 CPU 和内存的占用是非常大的。如果允许 IPSec 系统进行不必要的数据包加解密，那系统会很容易受到拒绝服务攻击。所以当需要加解密时，只有在数据包被成功认证后才进行加解密。

在这些步骤之后，如果数据包还未被抛弃，它将被提交到传输层协议或转发到目的 IP 地址域所指定的节点。

13.3.4 Internet 密钥交换

密钥管理是使用 IPSec 的关键，如通过密钥管理提供一种方法用于与其他人协商协议、加密算法以及在数据交换中使用的密钥。此外，IPSec 需要知晓实体之间的所有这样的协定。IETF 的 IPSec 工作组已经指定所有兼容的系统必须同时支持手工和自动的 SA 和密钥管理。下面是关于这些技术的简要描述。

(1) 手工：手工密钥和 SA 管理是最简单的密钥管理方式。一个人（通常是系统管理员）手工来配置每一个系统，提供与其他系统进行安全通信相关的密钥信息及密钥管理数据。手工技术可以在小范围、静态环境中有效作用，但是这种方法对于大型的网络并不适合。

(2) 自动：通过使用自动的密钥管理协议，可以创建 SA 所需要的密钥。自动管理同样为正在变化的较大型的分布式系统提供更大的可扩展性。对于自动管理，可以使用各种各样的协议，但是 Internet 密钥交换（IKE）似乎已经成为当前的工业标准。

IKE 协议不是一个单一的协议，它是两个协议的混合物，它将 Internet 安全关联和密钥管理协议与 Oakley 密钥交换协议集成在一起。

1. IKE 交换阶段

IKE 通过两个阶段来实现它的功能。在第一个阶段，两个 IKE 对等方使用一个普通的 IKE 安全关联为通信建立一个安全隧道。IKE 提供了 3 种交换密钥信息和建立 SA 的模

式,第一阶段仅仅使用主模式和野蛮模式。在第二个阶段,将为像 IPSec 这样的服务或者是任何其他需要密钥信息和参数协商的服务协商 SA。第二个阶段通过快速模式来完成。

2. IKE 交换模式

IKE 提供了 3 种交换密钥信息和建立 SA 的模式,即主模式、野蛮模式和快速模式。

(1) 主模式:IKE 的主模式为建立第一个阶段的 IKE SA 提供了一个 3 阶段机制,它用于协商以后的通信。在该模式中,实体间要协商足够多的东西(如认证和加密算法、Hash 函数和密钥),这样就能够进行足够长时间的安全通信以便建立起用于以后通信的 SA。在主模式中,要在 SA 发起者与响应者之间交换 3 个双向消息,如图 13.15 所示。在第一次交换中,两个实体协商基本的算法和 Hash 函数。在第二次交换中,它们交换用于 Diffie-Hellman 密钥交换协议的公钥并传递各自的 Nonce(为证明其身份而由另一个实体签名并返回的随机数)。在第三次交换中,它们验证身份信息。

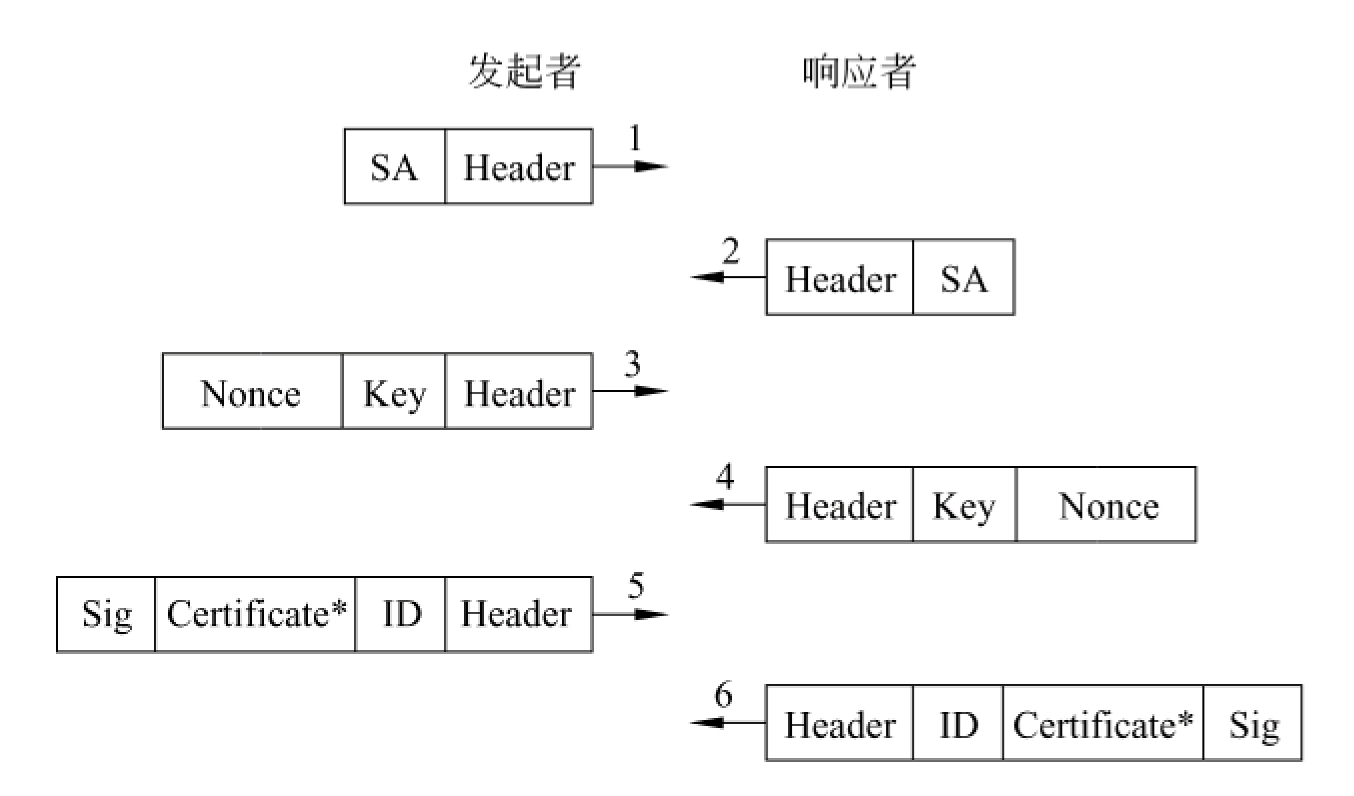


图 13.15 主模式中的消息交互

* 包含可选的证书载荷

(2) 野蛮模式:野蛮模式与主模式类似,也用于建立初始的 IKE SA。但野蛮模式在消息构建的方式上不同,从而将交换的数目从 3 个减为 2 个。在野蛮模式中,发起者在交换的开始生成一个 Diffie-Hellman 对,并对第一个数据包作尽可能多的实际工作:建议一个 SA,传递 Diffie-Hellman 公开值,发送给一个用于对方签名的一次性随机数,并发送一个 ID 数据包。这样借助于第三方,响应者可以用它来验证发起者的身份,参见图 13.16。然后,响应者发回为了完成本次交换而需要的一切消息。发起者剩下要做的事就是确认本次交换。野蛮模式的优点是速度快,但并不对通信实体提供身份保护。这意味着实体要在建立

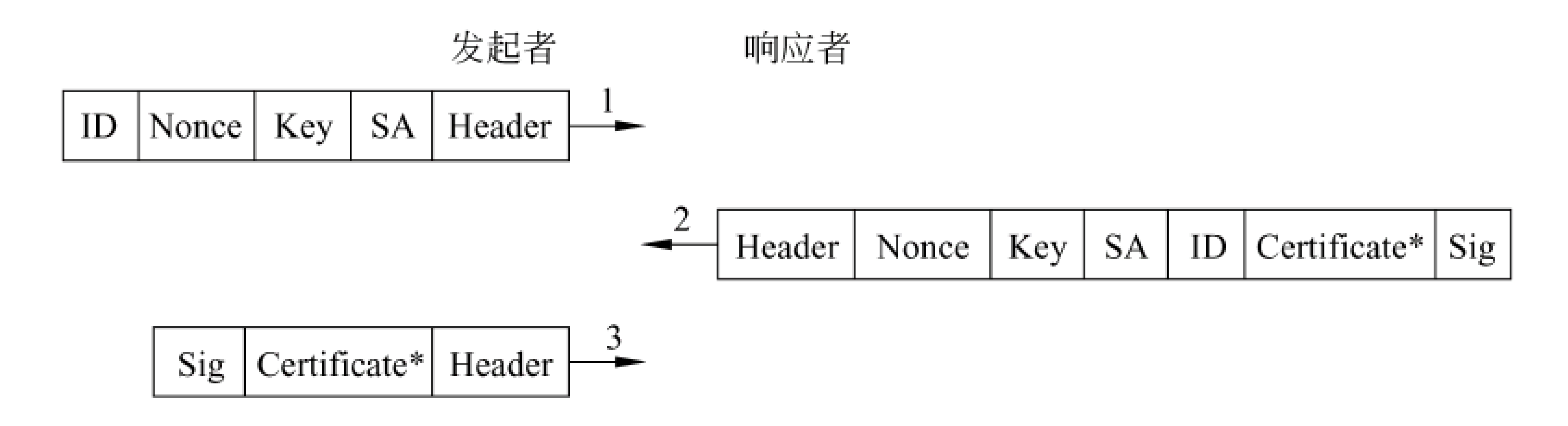


图 13.16 野蛮模式中的消息交互

* 包含可选的证书载荷

一个用于加密身份信息的安全 SA 之前交换身份信息。因此，监视一个用野蛮模式交换的人可以识别出刚刚产生了一个新 SA 的实体。

（3）快速模式：在两个通信实体使用主模式或者野蛮模式建立了一个 IKE SA 之后，它们就可以使用快速模式了。与主模式和野蛮模式不同的是，快速模式仅仅用于协商通用的 IPSec 安全服务并生成新的密钥信息。因为数据已经位于一个安全的隧道里面（每一个数据包都是加密的），在快速模式中可以提供更多一点的灵活性。快速模式中的数据包都是加密的，并且以一个杂凑载荷开始，该杂凑载荷是由一致认可的伪随机函数和为 IKE SA 而衍生出的认证密钥构成。杂凑载荷用于认证数据包的其余部分。快速模式定义了数据包的哪一部分将包括在杂凑中。如图 13.17 所示，发起者发送一个带有快速模式杂凑的数据包，该数据包中包含有建议和一次性随机数。紧接着响应者用一个类似的数据包来应答，此时响应者生成它自己的一次性随机数，并将发起者的一次性随机数包括在用于确认的快速模式杂凑中。然后，发起者发回一个对这两个一次性随机数的确认快速模式杂凑，从而完成本次交换。最后，将衍生出的密钥作为杂凑使用的密钥，两个实体将对下面各项的连接作杂凑运算：一次性随机数、SPI 以及发起该交换的 ISAKMP 头中的协议值。新计算出的杂凑作为该 SA 的新的口令字。

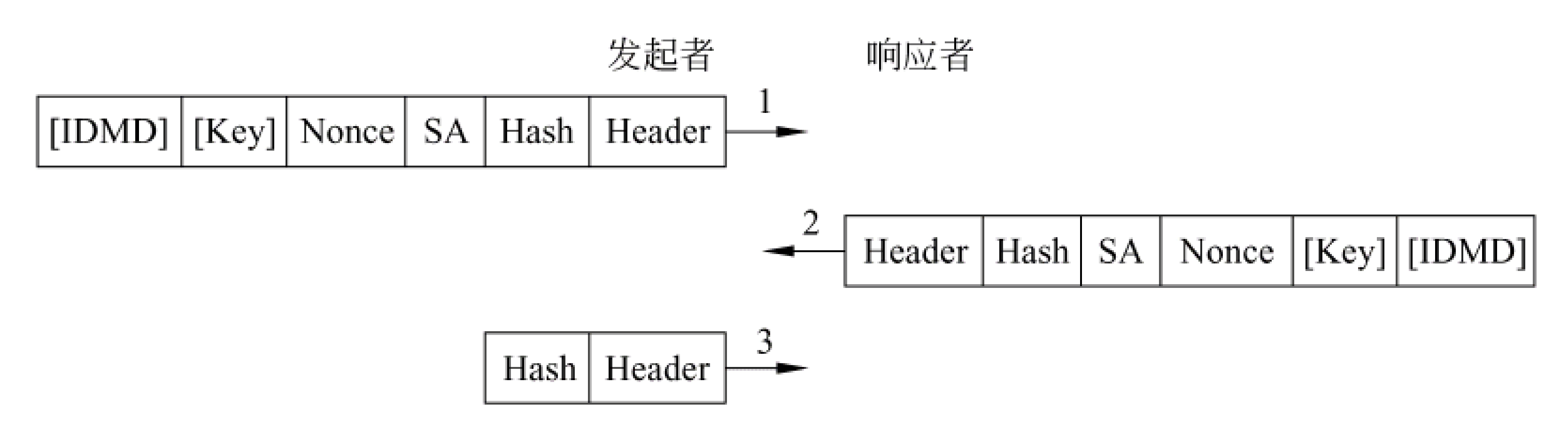


图 13.17　快速模式中的消息交互

13.4　TLS 协议

1994 年，Netscape 公司为了保护 Web 通信协议 HTTP/S-HTTP，开发了 SSL（Secure Socket Layer）协议。该协议第一个成熟的版本是 SSLv2.0，被集成到 Netscape 公司的 Internet 产品中，包括 Navigator 浏览器和 Web 服务器产品等。SSLv2.0 协议的出现，基本上解决了 Web 通信协议的安全问题。随后，Microsoft 公司对该协议进行了一些修改，发布了 PCT（Private Communication Technology）协议，并应用于 Internet Explorer 等产品中。1996 年，Netscape 公司发布了 SSLv3.0，该版本增加了对除了 RSA 算法之外的其他算法的支持和一些新的安全特性，并且修改了前一个版本中存在的一些安全缺陷，与 SSLv2.0 相比更加成熟和稳定，因此，很快成为事实上的工业标准。1997 年，IETF 基于 SSLv3.0 协议发布了 TLSv1.0（Transport Layer Security）传输层安全协议草案，实质上是 SSLv3.1，与 SSLv3.0 十分接近。Microsoft 公司也不再完善 PCT，而是与 Netscape 公司一起宣布支持该开放标准。1999 年，正式发布了 RFC2246。

TLS 协议是基于会话的加密和认证的 Internet 协议，它在两实体（客户和服务器）之间提供了一个安全的管道。为了防止客户-服务器应用中的监听、篡改及消息伪造，TLS 提供

了服务器认证服务和可选的客户端认证服务。通过在两个实体之间建立一个共享的秘密，TLS 提供了机密性服务。

TLS 工作在传输层（在应用层之下），并与使用的应用层协议无关。因此，应用层协议（如 HTTP、FTP 和TELNET）可以透明地置于 TLS 之上，图 13.18 显示了 TLS 在 TCP/IP 中的位置。

SMTP	HTTP	NNTP
TLS		
TCP		
IP		

图 13.18　TLS 在 TCP/IP 中的位置

TLS 握手协议	TLS 更改密码规范协议	TLS 警告协议	HTTP
TLS 记录协议			
TCP			
IP			

图 13.19　TLS 协议栈

13.4.1　TLS 体系结构

TLS 不是单个协议，而是两层协议，如图 13.19 所示。

TLS 记录协议为不同的更高层协议提供了基本的安全服务，特别是它可为 Web 客户/服务器的交互提供传输服务的超文本传输协议（HTTP）提供安全服务。下面的 3 个更高层协议也是 TLS 的组成部分：握手协议、更改密码规范协议和警告协议。

系统的状态描述了一个特定时间点的系统。状态变换是从一个状态到另一个状态的过程。一个特定对象的所有可能状态以及状态变换的组合称为一个状态机。TLS 有两个状态机：一个用于协议的客户端；另一个用于协议的服务器端。每一个端点必须实现协议的匹配方。状态机之间的交互作用称为握手。

协调客户端和服务器端的状态机是 TLS 握手协议的责任，这可以使得每一方的状态机持续地运行，尽管状态机并不是完全并行的。从逻辑上讲，状态要进行两次协调：第一次用于当前的运行状态，第二次用于未决的状态（在握手协议期间）。此外，要分别维护读状态和写状态。一个 TLS 会话可以包括多个安全的连接，实体间可以同时有多个会话。

TLS 规范中定义了一个会话状态的要素。

（1）会话标识符：由服务器选择的一个任意的字节序列，用于标识一个活动的或可重用的会话状态。

（2）对方的证书：对方的 X.509v3 证书。此要素可为空。

（3）压缩方法：在加密之前所使用的数据压缩算法。

（4）密码规范：指定数据加密算法（如空、DES 等）和用于消息认证的 MAC 算法（如 MD5 或者 SHA1）。它同样定义了诸如杂凑长度这样的密码属性。

（5）主秘密：在客户端和服务器之间共享的一个 48B 的秘密。

（6）可重用否：一个标志，用于指明该会话是否可以用来发起一个新的连接。

此外，TLS 规范中也定义了一个连接状态的要素。

（1）服务器和客户端随机数：由客户端和服务器独立选择的用于各自连接的字节

序列。

(2) 服务器写 MAC 秘密：用于对服务器写的数据进行 MAC 操作的秘密密钥。

(3) 客户端写 MAC 秘密：用于对客户端写的数据进行 MAC 操作的秘密密钥。

(4) 服务器写密钥：用于由服务器加密并由客户端解密的数据的对称密码密钥。

(5) 客户端写密钥：用于由客户端加密并由服务器解密的数据的对称密码密钥。

(6) 初始化向量：每一个使用 CBC 模式的分组密码，都需要一个初始化向量(IV)。该域首先由 TLS 握手协议对其初始化。此后，每一个记录最后的密文块将保留，以用于其后的记录。

(7) 序列号：对于每一个连接，每一方独立维护用于传输出的和接收到的消息的序列号。当一方发送或者接收到一个 ChangeCipherSpec 消息时，就将该适合的序列号置为 0。序列号的类型为 uint64，不超过 $2^{64}-1$。

13.4.2　TLS 记录协议

当要将数据传输到上层应用以及从上层应用中接收数据时，这些操作都是在 TLS 的记录层里来完成的，参见图 13.20，正是这一层对数据进行了加密、解密和认证。

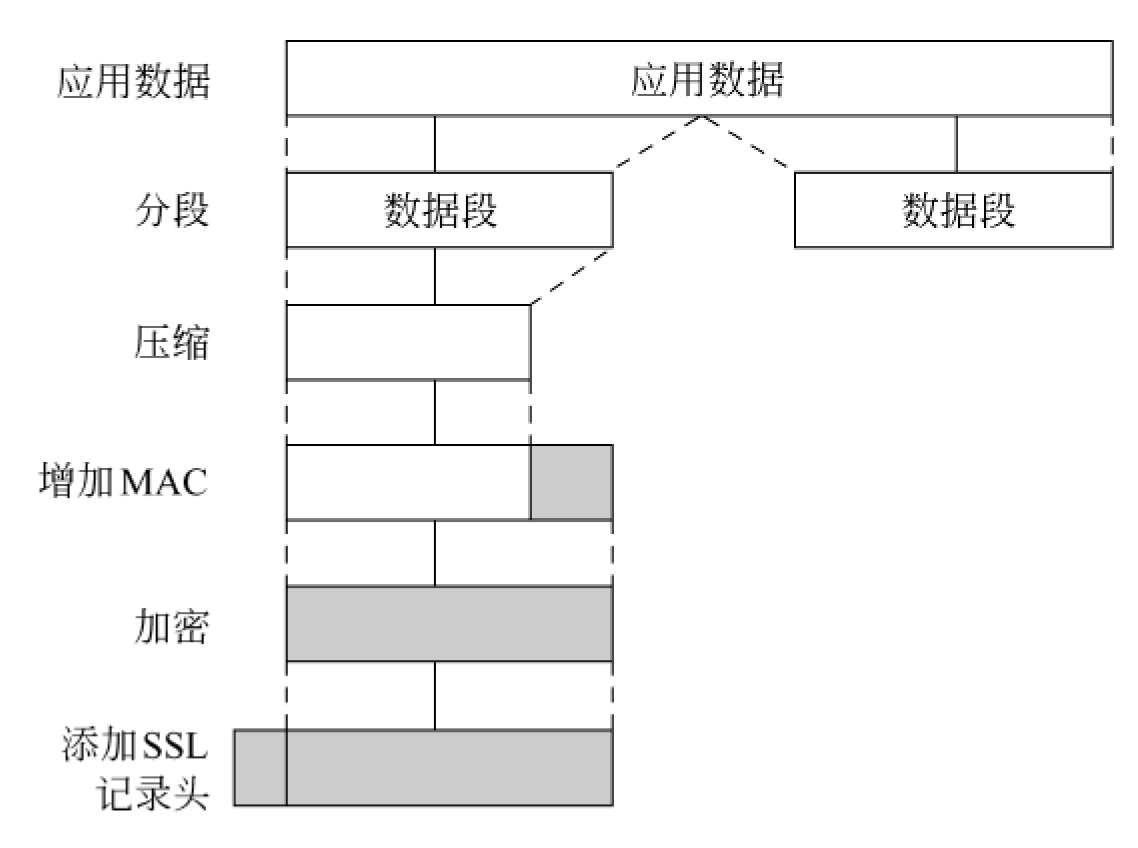


图 13.20　TLS 记录协议的操作过程

记录层主要执行下列 5 步操作。

(1) 当记录层从上面的应用层接收到不间断的数据流时，将对数据进行分段，或者是将其分割成可管理的明文块(也称记录)。每一个记录的长度为 16KB 或者更小。

(2) 作为一种选择，可使用当前会话状态中定义的压缩算法对明文块记录进行压缩。

(3) 对每一个明文记录，计算 MAC 值。为此目的，需使用先前建立的共享秘密。

(4) 使用先前达成共识的用于该会话的对称密码来对压缩的(或者明文)数据以及与之相关联的 MAC 加密。压缩必须是无损压缩，加密不能使记录的整个长度的增加超过 1024B。

(5) 将包含有下面各域的头作为一个前缀添加到每一个记录中。

① 内容类型：该域表明用于处理在下一个高层协议中被封装的记录的协议。

② 主版本：该域表明使用的 TLS 协议的主版本号，TLS 的值为 3。

③ 次版本：该域表明使用的 TLS 协议的次版本号，TLS 的值为 1。

④ 压缩的长度：该域表明以字节计的明文记录的总长度。

接收到这些信息的实体要将该过程逆置，也就是说，简单地将解密和认证函数反过来执行。

图 13.21 显示了 TLS 记录头的格式。



图 13.21　TLS 记录头的格式

13.4.3　TLS 更改密码规范协议和警告协议

1. 更改密码规范协议

设计更改密码规范协议的目的是为了表示密码策略的变化。该协议包括一个单一的消息，该消息由值为 1 的单个字节组成，它告知记录层按照当前密码规范中所指定的方式进行加密和压缩。在完成握手协议之前，客户端和服务器都要发送这一消息，以便通知对方其后的记录将用刚刚协商的密码规范以及相关联的密钥来保护。一个意外的更改密码规范消息将生成一个"意外消息"警告。

2. 警告协议

TLS 记录层支持的内容类型之一是警告类型。警告协议将警告消息及其严重程度传递给 TLS 会话中的主体，它的每个消息由两个字节组成，第 1 个字节表示警告的级别，其值为 1(表示警告)或 2(表示致命的警告)，第 2 个字节包含了指出特定警告的代码。就像由记录层处理的应用层数据一样，警告消息也用当前连接状态所指定的方式来压缩和加密。

当任何一方检测到一个错误时，检测的一方就向另一方发送一个消息。如果警告消息有一个致命的后果，通信的双方立即关闭连接。双方都需要忘记任何与该失败的连接相关联的会话标识符、密钥和秘密。对于所有的非致命的错误，双方可以缓存信息以重用该连接。

下面的错误警告是致命的。

(1) 意外消息(Unexpected_message)：如果收到一个不合适的消息，就返回该消息。

(2) 错误的记录 MAC(Bad_record_mac)：如果收到一个带有不正确消息认证码的记录，就返回该消息。

(3) 解密失败(Decryption_failed)：对密文解密不合法，或者密文不是分组长度的偶数倍，或者在检查时发现它的填充值不正确。

(4) 解压缩失败(Decompression_failure)：如果解压缩函数收到一个不正确的输入(如数据不能解压缩或者数据解压缩至一个超出长度)，将返回该消息。

(5) 握手失败(Handshake_failure)：返回该消息表明在给定的可用选项下，发送者无法协商一个可以接受的安全参数集。

(6) 非法参数(Illegal_parameter):握手中的某数据域超出范围或者与其他的数据域不一致。

(7) 记录溢出(Record_overflow):收到的 TLS 记录的载荷(密文)长度超过了(2^{14}+2048)B,或者密文解密后的长度大于(2^{14}+1024)B。

(8) 未知 CA(Unknown_ca):收到了合法的证书链或部分证书链,但无法接受证书,因为 CA 证书不能被定位或者不能和已知的、可信的 CA 相匹配。

(9) 访问拒绝(Access_denied):收到了合法的证书,但当应用于访问控制时,发送者决定不再继续执行该协商。

(10) 译码错误(Decode_error):消息不能被译码,因为某些字段超出了指定的范围,或者消息的长度不正确。

(11) 输出限制(Export_restriction):检测出协商与输出限制不兼容,如企图为 RSA_EXPORT 握手方法传输一个 1024b 的临时 RSA 密钥。

(12) 协议版本(Protocol_version):客户企图协商的协议版本被识别出,但不被支持,例如,由于安全原因,旧的协议版本已被避免使用。

(13) 不充分安全(Insufficient_security):当由于服务器要求比客户支持的更安全的密码而使得协商失败时,返回该警告而不是"握手失败"警告。

(14) 内部错误(Internal_error):与对等实体或协议的正确性无关的内部错误使得协商不能继续。

其余的警告如下。

(1) 错误证书(Bad_certificate):返回该消息表明证书已损坏(也就是,它包含有一个无法通过验证的签名)。

(2) 不支持证书(Unsupported_certificate):返回该消息表明证书是一个不被支持的类型的证书。

(3) 证书吊销(Certificate_revoked):返回该消息表明证书已被其签发者吊销。

(4) 证书过期(Certificate_expired):返回该消息表明证书已经过期。

(5) 证书未知(Certificate_unknown):返回该消息表明在处理证书的过程中出现了一些其他(未指明)的问题,这是不可接受的。

(6) 关闭通知(Close_notify):该消息通知接收者,发送者将不再通过该连接发送任何信息。在关闭一个连接的写一方之前,每一方都需要发送该消息。

(7) 解密错误(Decrypt_error):握手加密操作失败,包括不能正确地验证签名,解密密钥改变或者不能确定所完成的消息的合法性。

(8) 用户中止(User_canceled):这个握手由于某个与协议失败无关的原因而被中止。

(9) 不能重新协商(No_renegotiation):在初始握手之后,客户发送的作为 hello 请求的响应或者服务器发送的作为客户 hello 的响应的警告。

13.4.4 TLS 握手协议

TLS 握手协议建立连接会话状态的密码参数,该过程在 TLS 记录协议之上进行。当 TLS 协议的客户和服务器开始第一次通信时,首先需要协商协议版本,选择密码算法,相互进行认证(可选功能),并使用公钥密码技术生成共享秘密。

TLS 握手协议包括以下步骤。

(1) 交换 hello 消息以协商密码算法,交换随机值并检查会话是否可重用。

(2) 交换必要的密码参数,使客户和服务器能够协商预主秘密 PremasterSecret。

(3) 交换证书和密码信息,使客户和服务器能够进行相互认证。

(4) 使用交换的随机值和 PremasterSecret 生成主秘密 MasterSecret。

(5) 为记录协议提供安全参数。

(6) 允许客户和服务器校验对方是否计算出了相同的安全参数,以及校验上述握手过程是否被攻击者窃听。

1. 握手流程

握手协议的过程描述如下。

客户向服务器发送 ClientHello 消息,服务器回应 ServerHello 消息。ClientHello 和 ServerHello 消息建立下列安全属性:协议版本、会话 ID、CipherSuite 和压缩方法,同时生成并交换两个随机数:ClientHellorandom 和 ServerHellorandom。

实际的密钥交换使用 4 条消息:ServerCertificate、ServerKeyExchange、ClientCertificate 和 ClientKeyExchange。

在 hello 消息之后,如果需要认证服务器的话,服务器将发送其证书。另外,如果需要的话,还要发送 ServerKeyExchange 消息。对服务器进行认证之后,服务器可以请求客户证书。然后,服务器将发送 ServerHelloDone 消息,指示握手协议的 hello 消息阶段结束,服务器等待客户的响应。如果服务器发送了 CertificateRequest 消息,客户必须发送客户证书,然后发送 ClientKeyExchange 消息,消息的内容取决于 ClientHello 和 ServerHello 定义的密钥交换算法。如果客户发送了具有签名能力的证书,则需要发送 CertificateVerify 消息显式地校验该证书。

随后,客户发送 ChangeCipherSpec 消息,并将未决(pending)的 CipherSpec 复制为当前的 CipherSpec。然后,客户在新的算法、对称密钥和 MAC 秘密之下立即发送 finish 消息。服务器响应客户的消息,发送其 ChangeCipherSpec 消息和 finish 消息。到此为止,握手结束,客户和服务器可以开始发送应用层数据。握手流程如图 13.22 所示。

如果客户和服务器决定重用以前的会话,则过程如下。

客户使用要重用的会话的会话 ID 发送 ClientHello 消息,服务器在会话缓存中检查该 ID,如果找到匹配的 ID,并且服务器同意在这样的条件下建立连接,它将发送包含相同 ID 的 ServerHello 消息。此时,客户和服务器都必须在发送 finish 消息之前发送 ChangeCipherSpec 消息,流程如图 13.23 所示。如果服务器没有找到相匹配的会话 ID,服务器将生成新的会话 ID,双方需要进行完整的握手。

2. 基本消息描述

(1) Hello 消息:服务器和客户使用 Hello 消息来交换安全相关的信息,如随机数、CipherSuite 等,包括 ClientHello 消息、ServerHello 消息和 HelloRequest 消息。

(2) ServerCertificate 消息:该消息表示服务器发送证书,在 ServerHello 消息之后立即被发送。证书的类型必须与所选择的 CipherSuite 中的密钥交换算法相匹配,一般情况下是 X.509v3 格式的证书,证书中要包含与密钥交换算法相匹配的密钥。

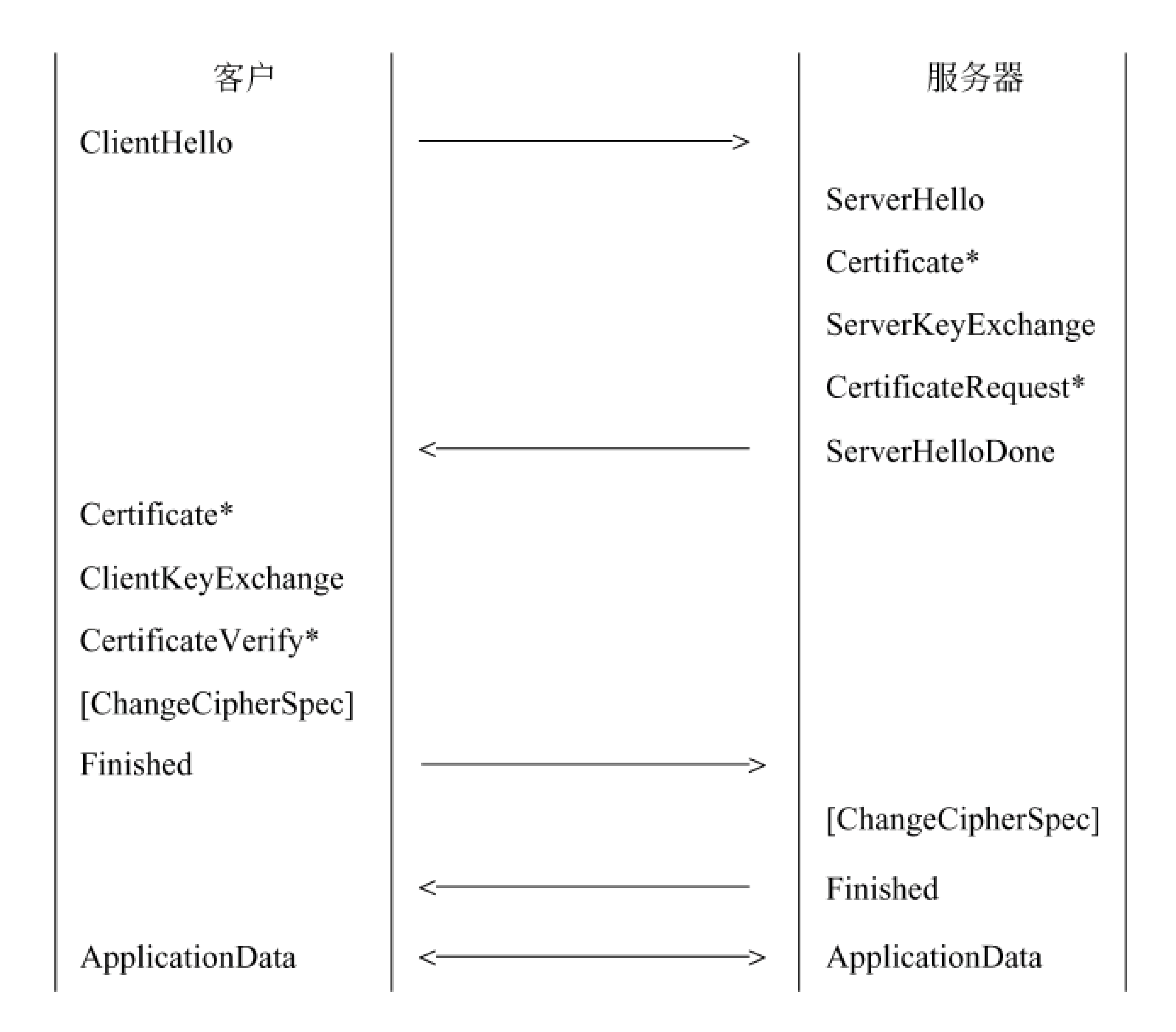


图 13.22　握手过程消息流程图

* 可选的消息或者根据具体情况来决定是否要发送的消息

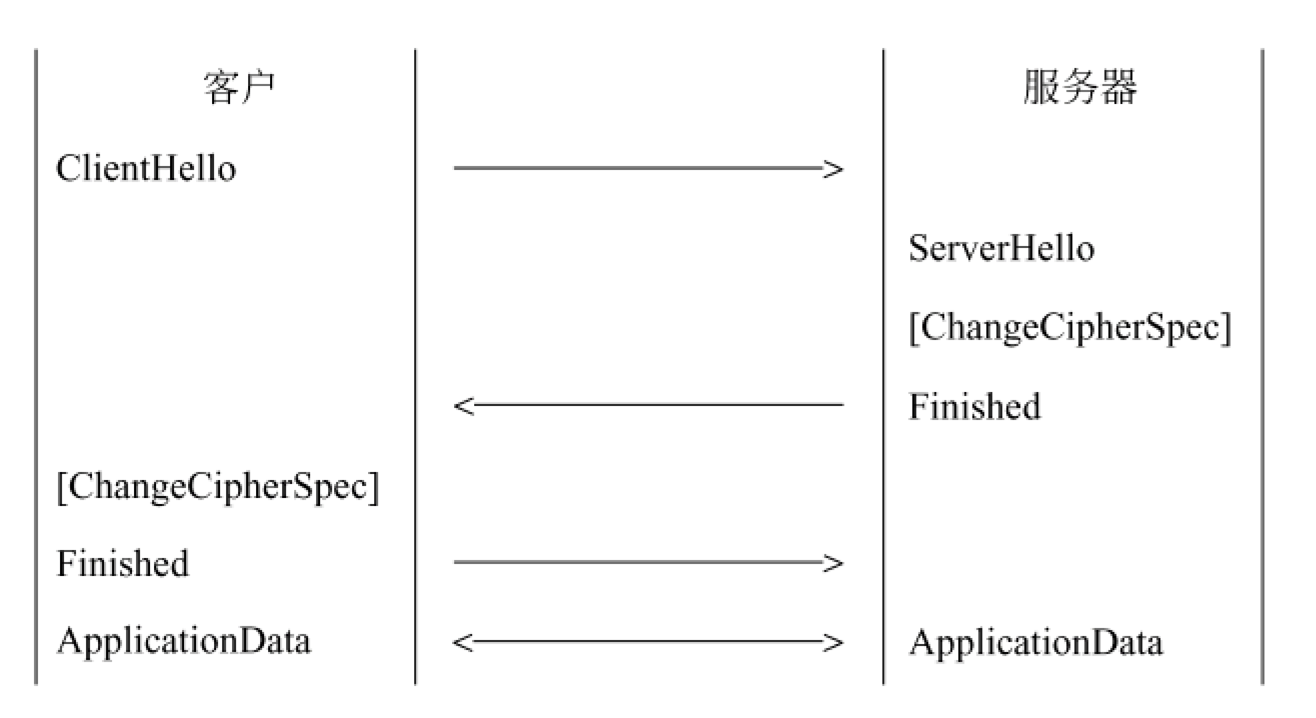


图 13.23　会话重用的握手过程消息流程图

(3) ServerKeyExchange 消息：该消息仅在服务器发送的 ServerCertificate 消息中没有包含足够的信息使客户可以交换 PremasterSecret 时发送，为客户端协商 PremasterSecret 传递密码信息。

(4) CertificateRequest 消息：服务器发送该消息指示客户提供其证书。

(5) ServerHelloDone 消息：服务器发送该消息指示 Hello 阶段结束。

(6) ClientCertificate 消息：这是客户接收到 ServerHelloDone 消息之后能够发送的第一条消息，该消息只有在服务器要求证书的情况下才发送。如果客户没有合适的证书，也可以发送不包含证书的 ClientCertificate 消息。

(7) ClientKeyExchange 消息：该消息由客户发送，一般使用 RSA 公钥加密的方式直接传输 PremasterSecret，或者使用 Diffie-Hellman 方法使双方商定该秘密。

(8) CertificateVerify 消息：该消息用来提供显式的客户证书校验。

(9) Finished 消息：该消息在 ChangeCipherSpec 消息之后发送，用来校验密钥交换和认证过程是否成功。Finished 消息是第一个使用刚刚商定的算法、密钥和秘密进行保护的消息。该消息的接收者必须校验消息内容的正确性。

13.4.5 TLS 密码特性

记录层主要使用了对称密码算法、MAC 算法；握手层主要使用了基于公开密钥技术的密钥交换算法（如 RSA、临时或匿名 Diffie-Hellman），其密码特性是 TLS 协议的重要特点。

1. 伪随机数生成

TLS 协议的记录协议和握手协议中许多地方都需要使用加密的 MAC，即用秘密保护的消息摘要。在不知道 MAC 秘密的情况下伪造 MAC 是不可能的。协议中使用的方法称为 HMAC 算法。

HMAC 可以使用多种不同的 Hash 算法。TLS 中使用了两种不同的 Hash 算法，即 MD5 和 SHA-1，分别表示为 HMAC_MD5(secret, data) 和 HMAC_SHA(secret, data)。另外，协议中还使用了一种秘密扩展算法，用来将秘密扩展为用于密钥生成的密钥材料。该伪随机函数(PRF)把秘密、种子和识别符作为输入，产生任意长度的输出。为保证 PRF 尽可能地安全，使用了两种 Hash 算法，这样，只要有一种算法安全，就可以确保该 PRF 的安全性。其计算过程如下。

```
P_hash(secret,seed)=HMAC_hash(secret,A(1)||seed)||
                    HMAC_hash(secret,A(2)||seed)||
                    HMAC_hash(secret,A(3)||seed)||…
```

这里的 ‖ 表示串联。A()定义如下：

```
A(0)=seed
A(i)=HMAC_hash(secret,A(i-1))
```

可以多次重复使用 P_hash，直到产生足够长的数据。

TLS 协议中的 PRF 是这样构造的：将秘密分成两半，一半使用 P_MD5 产生数据，另一半使用 P_SHA-1 来产生数据，然后将两个函数的输出进行异或操作。S1 和 S2 是等长的两半秘密，S1 是秘密的前一部分，S2 是秘密的后一部分。其长度通过对初始秘密的长度除 2 进位取整实现，这样，如果秘密的长度为一个奇数，则 S1 的最后一个字节与 S2 的第一个字节相同。

L_S＝以字节计的秘密的长度。

```
L_S1=L_S2=ceil(L_S/2)
```

秘密按上述方式分为两半（可能共享一个字节），S1 为前 L_S1 个字节，S2 为后 L_S2 个字节，则 PRF 的定义如下。

```
PRF(secret,label,seed)=P_MD5(S1,label||seed) XOR P_SHA-1(S2,label||seed)
```

2. 密钥计算

协议中使用的主秘密(Master Secret)被扩展为安全的字节序列，并分配给当前连接状

态所需的 MAC 秘密、对称密钥和初始化向量 IV。生成 MAC 秘密和对称密钥时，主秘密用作熵源，为密钥生成算法提供新鲜的密钥材料和初始化向量。

密钥材料的生成算法如下。

```
key_block=PRF(SecurityParameters.master_secret,"key expansion",
          SecurityParameters.server_random||SecurityParameters.client_random);
```

直到产生足够的输出。然后进行分段处理：

```
client_write_MAC_secret[SecurityParameters.hash_size]
server_write_MAC_secret[SecurityParameters.hash_size]
client_write_key[SecurityParameters.key_material_length]
server_write_key[SecurityParameters.key_material_length]
client_write_IV[SecurityParameters.IV_size]
server_write_IV[SecurityParameters.IV_size]
```

对于所有的密钥交换算法，使用相同的算法将 pre_master_secret 转换成 master_secret。

```
master_secret=PRF(pre_master_secret,"master secret",
              ClientHello.random||ServerHello.random);
```

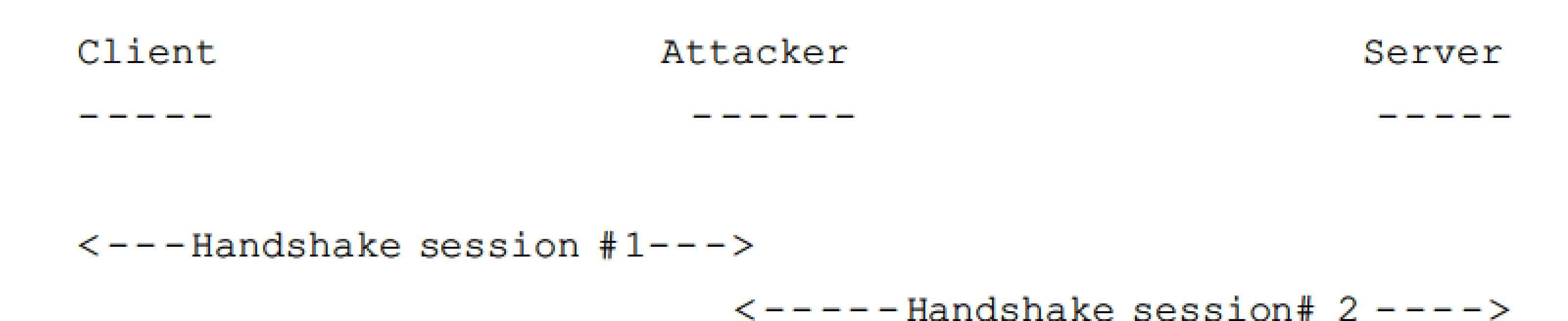
算法进行到产生了 48 个字节的伪随机输出为止。

13.5　TLS/SSL 重协商安全扩展

TLS/SSL 容易遭受一种新的重协商攻击：攻击者干扰客户端和服务器之间的 TLS 会话，并插入其任意选区的数据。客户端认为已经和服务器建立了一个 TLS 连接，而服务器则认为攻击者和客户端是同一个实体，从而把攻击者传送的数据当作是客户端发送的而接受。这一攻击可能导致多种潜在的安全威胁。我们提出了 TLS/SSL 的安全重协商扩展方案，使得客户端和服务器能够显式地区分 TLS/SSL 重协商会话与初始协商会话，从而防止该攻击。

13.5.1　Ray 的攻击方法介绍

TLS 允许客户端或者服务器在 TLS 会话中发起重协商的请求，建立新的密码参数。重协商握手是在初始协商的基础上进行的，并受到了初始协商的保护。然而，Ray 指出[7]：在客户端和服务器之间的重协商存在鉴别不一致问题，从而导致双方在重协商和初始协商之间产生匹配错误。这一安全缺陷使得攻击者可以拦截客户端的传输层连接，插入其自己选择的数据，作为客户端与服务器之间交互通信的 prefix。该攻击[7,8]可以描述如下：



```
Client                    Attacker                        Server
-----                      ------                          -----

<---Handshake session #1--->
                            <-----Handshake session# 2 ---->
```

```
                                             <=====Attack injection ======>
                                             <===Renegotiation triggered===>
<------------------------Handshake ========================>
<================Client-Server Traffic===================>
```

在该攻击中,攻击者拦截了客户端的握手请求,并不立即转发给服务器。敌手与服务器建立一个 TLS 连接,并发送其自己选择的应用层命令。当重协商被触发(由敌手或者服务器)之后,攻击者允许客户端继续其与服务器的 TLS 握手。该握手对于客户端来说是明文的,但对于服务器来说是经过攻击者的 TLS 连接加密的。在客户端和服务器之间的握手完成之后,他们之间的通信就由其新建立的安全参数进行保护,敌手并不能读取其通信。然而,服务器认为它与攻击者之间进行的通信和与客户端之间进行的通信是来自于同一个实体,从而接受攻击者插入的数据作为之后与客户端之间传输数据的前置数据(prefix)。

SSLv3 同样允许客户端或者服务器在会话中发起重协商的请求,正如 TLS 一样,从而也容易遭受这一类攻击。针对 TLS 和 SSL 的攻击可能直接影响到 HTTPS、IMAP 及 SMTP 等众多协议[9],导致许多潜在的安全威胁[8]。

13.5.2 安全重协商的扩展方法

对于 13.5.1 小节介绍的攻击来说,如果客户端和服务器能够显式地区分 TLS 会话中的重协商与初始协商,就可以有效地防止。本节定义了一种 TLS 安全重协商的扩展方法,在密码学意义下区分了重协商与初始协商。初始协商则保持不变,尽可能地与 RFC5246 兼容。如果重协商被触发,客户端和服务器分别向其对等实体确认其已经知道了该过程是重协商连接,从而不会导致握手匹配错误。根据这种扩展方式,上述攻击将被检测到并有效地阻止。本节定义的安全的重协商简单易实现,不会带来附加的存储代价,而且可以直接应用于 SSL 协议实现安全重协商。

本节使用的 MUST、SHOULD、SHOULD NOT 等术语遵循 RFC2119[10]的解释。首先给出 TLS 协议中 Finished 消息的规范定义,然后提出安全的重协商的方法。

1. TLS 协议中规定的 Finished 消息

在 TLS 协议[11]中规定了一个 Finished 消息,在 ChangeCipherSpec 之后立即发送 Finished 消息,用于验证密钥交换和鉴别过程是否成功。Finished 消息使用了刚协商出的算法、密钥、秘密进行保护。Finished 的接收方 MUST 验证其内容是正确的。客户端/服务器在发送了自己的 Finished 消息、接收到对方的 Finished 消息并进行验证之后,就可以通过建立起来的安全连接发送和接收应用数据。

Finished 消息的结构:

```
    struct {
        opaque verify_data[verify_data_length];
            } Finished;
    verify_data
    PRF(master_secret, finished_label,
Hash(handshake_messages))
              [0..verify_data_length- 1];
```

其中，master_secret 是用于产生客户端和服务器的 MAC 密钥和加密密钥的主秘密，master_secret 也用于 resume 之前的会话。

2. 重协商的安全扩展

这里规定了一种新的扩展的 Finished 消息，从密码学意义下把重协商与实体之间进行的 TLS 会话进行绑定，以实现 TLS 安全的重协商。

扩展的 Finished 消息的结构与上述中的定义在形式上完全一致：

```
  struct {
          opaque verify_data[verify_data_length];
           } Finished;
    verify_data
PRF(Hash(master_secret, initial_master_secret),
    finished_label, Hash(handshake_messages))
             [0..verify_data_length-1];
```

其中，master_secret 是当前进行的重协商握手的主秘密，initial_master_secret 是客户端和服务器在此之前进行的初始协商握手中协商出来的主秘密。

3. 关于初始协商握手

上述规定的扩展 Finished 消息只是针对 TLS 重协商握手而定义的，对于 TLS 初始协商握手则不作任何变动。因此，客户端和服务器进行初始协商的实现方式与 TLS[11] 中的规范是完全一样的。

4. 关于重协商握手

如果客户端和服务器之间进行重协商握手，则 MUST 按照以下方式验证 Finished 消息：

客户端/服务器检查接收到的 Finished 消息中的 verify_data 字段，并且验证其是否等于 PRF(Hash(master_secret，initial_master_secret)，finished_label，Hash(handshake_messages))。如果二者相同，客户端/服务器就知道了该消息是扩展的 Finished 消息，从而可以安全地使用重协商建立的参数，无需担心前面的攻击。如果接收到的 Finished 消息与其计算的扩展 Finished 消息不匹配，则重协商失败。

这里需要指出的是，在 SSL 协议[12] 中，Finished 消息的定义方式与 TLS 协议中的定义方式类似。因此，上述规定的安全重协商扩展方法可以直接用于 SSL 协议来抵抗 Ray 攻击。

5. 兼容性考虑

一般地，已经广泛部署的现有实现需要与支持安全重协商扩展的新实现互操作。这里考虑向下兼容性。

如果客户端和服务器只是进行 TLS 初始协商，那么升级后的实现与现有实现是完美兼容的。

在实际应用中，升级单个服务器比升级千万个向该服务器请求服务的客户端更容易操作。考虑支持扩展 Finished 消息的升级的服务器。如果双方触发了一次重协商，从客户端接收的 Finished 消息与扩展的 Finished 消息不匹配，表明客户端不支持这一扩展。如果要

确保上面的攻击不存在，服务器 MUST 立即取消该会话。否则，服务器可以按照 TLS 规范来验证 Finished 消息，即验证 verify_data 字段是否等于 PRF(master_secret，finished_label，Hash handshake_messages)。如果二者匹配，服务器知道客户端不支持扩展的 Finished 消息，如果为了兼容性，可以继续该会话，否则立即取消会话。

如果服务器没有进行升级来支持上述扩展，那么对于客户端来说，重协商的安全性与最大互操作性是难以同时确保的，因此需要准备其他的方式来进行处理。

6. 安全性考虑

使用本节规定的扩展的 Finished 消息来进行重协商，只要 master_secret 没有暴露，并且杂凑操作是安全的，则针对 TLS/SSL 的上述攻击即可有效地防止。TLS 服务器 SHOULD 进行升级以支持该扩展，并且 SHOULD NOT 允许客户端不使用该扩展而进行重协商。

13.6 小结

本章简要介绍了两种常用的典型分布式认证协议(Kerberos 协议和 X. 509 协议)，以及两类重要的网络安全通信协议(IPSec 协议和 TLS 协议)。在很多教科书中都介绍了这 4 种协议，因此，在写作过程中尽量引用现有教科书中表述比较好的素材。另外，我们的目的也是介绍这些协议的基本思想，因此，阐述得还不够细致。有关 Kerberos 协议和 X. 509 协议的发展历史和研究进展可分别参阅文献[1]和[2]，有关 IPSec 协议和 TLS 协议的发展历史和研究进展可分别参阅文献[3]和[4]。关于 IPSec 协议的描述也可参阅文献[5]，关于 SSLv3. 0 的描述也可参阅文献[6]和[12]。

参 考 文 献

[1] http://en. winkipedia. org/wiki/Kerberos_(protocol).
[2] http://en. winkipedia. org/wiki/X. 509.
[3] http://en. winkipedia. org/wiki/IPsec.
[4] http://en. winkipedia. org/wiki/Transport_Layer_Security.
[5] Carlton R. Davis IPSec:Securing VPNs，McGraw-Hill，2001.
[6] Burnett S，Paine S. RSA Security Official Guide to Cryptography. McGraw-Hill，2001.
[7] Ray M. Renegotiating TLS，November 2009.
[8] G-SEC，TLS & SSLv3 renegotiation vulnerability explained，November 2009，http:www. g-sec. lu/practicaltls. pdf.
[9] Rescorla E，Ray M，Dispensa S，Oskov N. Transport Layer Security(TLS)Renegotiation Indication Extension，draft-ietf-tls-renegotiation-03. txt.
[10] Bradner S. Key words for use in RFCs to Indicate Requirement Levels，BCP 14，RFC 2119，March 1997.
[11] Dierks T，Rescorla E. The Transport Layer Security(TLS) Protocol Version 1. 2，RFC 5246，August 2008.
[12] Freier A，Karlton P，Kocher P. The SSL Protocol Version 3. 0，http://www. mozilla. org/projects/security/pki/nss/ssl/draft302. txt，November 1996.

第 14 章　入侵容忍 CA 协议

高安全级别 PKI 系统必须考虑许多威胁，包括内部人员犯罪和系统木马攻击。本章主要介绍典型的和我们自主提出的一些入侵容忍 CA 协议，该类协议是构建高安全级别 PKI 系统的核心基础。入侵容忍 CA 协议的根本目的是为了保护 CA 系统的安全。在一般的网络系统中，为了防止入侵，通常是安装一个入侵检测系统来监视其行为，当发现一个非正常的系统行为时，入侵检测系统就会通知管理员，以采取适当的应对措施。而入侵容忍思想却是从一个完全不同的角度考虑问题，并不是阻止入侵或者在入侵发生后检测出来，而是尽可能把入侵所能造成的损失降到最低。其方法是确保一个攻击者即使能够攻破系统的某些部件（也称组件），却不能危及整个系统的安全性。要能危及整个系统的安全性，攻击者攻破的系统部件就必须达到足够的数量，而如此大规模的入侵通常要比单个的入侵容易发现得多。

从更严格意义上来说，入侵容忍 CA 协议的核心目的在于保证 CA 私钥的安全。假设 CA 系统只有一台服务器，那么 CA 私钥就必须存放在这台服务器上，攻击者一旦攻陷这台服务器，就可以轻松地获得 CA 私钥，从而冒充 CA。因此入侵容忍 CA 协议就将 CA 私钥分拆成若干个部分，每个部分存放在一台服务器上，保证了即使某些服务器被攻击者控制，CA 私钥仍然是安全的。为了达到这个目的，CA 私钥应该在其整个生命周期中都不能被某个特权主体所掌握，任何系统部件在任何时间都只能获得 CA 私钥的部分信息。在保护私钥的基础上，入侵容忍 CA 协议进一步的目的是在系统某些部件失去控制后，剩余的安全部件只要达到足够数量，就可以保证系统正常工作。这个思想的基础就是保证系统足够的冗余度，每个部件都需要保存多份部分私钥信息，以使得任意的部件组合（只要数量足够）都可以完成系统工作。

从以上分析可以看出，入侵容忍 CA 协议的研究焦点主要集中在以下几个方面。

(1) 如何分拆 CA 私钥，使得只有足够数量的部件联合才能恢复 CA 私钥。

(2) 如何保证 CA 私钥在整个生命周期中都是分散在多个服务器上，而不是由某个特权主体所掌握。

(3) 如何保证在系统的某些部件失去控制后，系统仍然能正常运行。

(4) 如何使系统可以快速地检测出发生错误的部件。

(5) 如何加强每个部件上部分私钥信息的安全性。

14.1　ITTC 入侵容忍 CA 协议

ITTC 入侵容忍 CA 协议[1]是由美国斯坦福大学 ITTC（Intrusion Tolerance via Threshold Cryptography）项目提出并实现的一个基于 RSA 的入侵容忍 CA 协议，该协议具有结构简单、安全性容易证明等特点，其主要思想是将 RSA 私钥拆分成 k 个数之和，然后分配给服务器。ITTC 项目是美国 DARPA 计划资助的一个项目。图 14.1 描述了 ITTC 入侵容忍 CA 协议的基本结构原理。

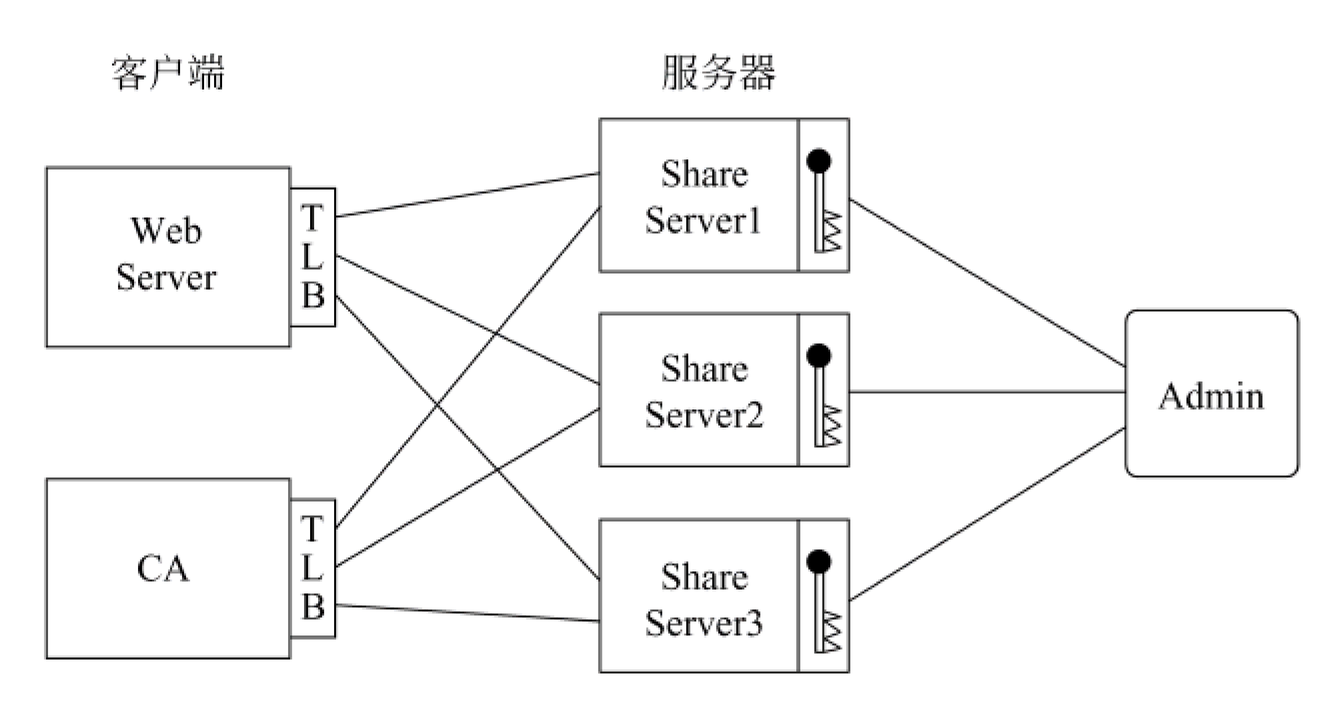


图 14.1　ITTC 入侵容忍 CA 协议的基本结构原理

ITTC 入侵容忍 CA 协议主要包括以下 4 个部件。

(1) Share Server：Share Server 掌管着 CA 私钥的部分秘密分享信息，即部分私钥。系统设计将 CA 私钥拆分到多个 Share Server 上，每个 Share Server 也可能掌管着多种分拆结构的部分私钥。

(2) Client：客户端，向 Share Server 发出签名或解密请求的机器。

(3) Administrator：管理部件，可以管理 Share Server 上不同的密钥，可以关闭或者暂时挂起 Share Server，或者在某些 Share Server 被攻击的情况下，通知其他部件采取行动等。

(4) Monitor：监视部件，或者说是一个审计部件，可以记录系统进行的操作，并转化为可为管理员所阅读的界面或文档。

14.1.1　密钥管理与签名

密钥管理的关键部分是如何在多个 Share Server 间分享 CA 的 RSA 私钥，使得这些 Share Server 可以联合完成 CA 工作，而无需重构 CA 私钥。

基本思想是将 CA 私钥拆分成多个部分，每个部分保存在一个 Share Server 上，拆分的基本思想基于文献[2]。假设要在 3 台 Share Server 间分享 CA 私钥，RSA 参数为 $K=(N,p,q,e,d)$，则拆分方案如下。

(1) 随机选取 3 个整数 d_1,d_2,d_3，满足 $-N\leqslant d_i\leqslant N(i=1,2,3)$，且 $d_1+d_2+d_3=d$。

(2) 将 d_1,d_2,d_3 分别保存在不同的 Share Server 上。

这样，一个攻击者即使攻破了两台 Share Server，也不可能获得 CA 私钥 d，只有 3 台联合才可能获得 d。

当客户端需要对消息 M 进行签名时：

(1) 客户端发送消息 M 给所有的 3 台 Share Server。

(2) 每台 Share Server 都可以计算部分签名 $R_i=M^{d_i} \bmod N$，并把 R_i 发送给客户端。

(3) 客户端在获得 3 个 R_i 后，计算下式：

$$R=R_1\cdot R_2\cdot R_3=M^{d_1}\cdot M^{d_2}\cdot M^{d_3}=M^{d_1+d_2+d_3}=M^d \bmod N$$

R 正是最终签名结果，可以看出在整个计算过程中，CA 私钥 d 并没有被重构，也没有被泄露的危险。

在上述基本方案中，只有所有的 Share Server 被攻破，才可能泄露 CA 私钥，但是一旦

失去某个 Share Server，系统也就无法正常工作，甚至完全丢失 CA 私钥，这一点还未达到入侵容忍的要求。因此，ITTC 将其扩展成一个(t,k)门限方案，即在总共 k 个 Share Server 中，至少 t 个 Share Server 就可以完成签名，或者重构私钥。例如，如果使用一个(3,4)门限方案，则即使失去一台 Share Server，系统仍可正常工作，而且攻击者控制的 Share Server 只要不到 3 台，也仍然无法获得 CA 私钥。可以看出，上述基本方案实际上是一个(k,k)门限方案。

ITTC 方案使用了一种组合的方法来实现门限思想，即使用多个(t,t)门限方案的组合，来完成(t,k)门限方案。

图 14.2 是两个门限方案的例子，每台服务器都保存着多个 d_i，每个 d_i 仍然是满足条件的随机数。可以看出，图 14.2(a)所示为任意 3 台服务器都能重构私钥 d，图 14.2(b)所示是任意 2 台服务器就能重构私钥 d。因此，也就能通过类似基本方案的步骤完成 CA 签名。基于这种方法以及文献[3]和[4]中的思想，ITTC 方案中构造了以下的门限方案。

$d=d_1+d_2+d_3$
$d=d_4+d_5+d_6$

Server1	Server2	Server3	Server4
d_1 d_4	d_2 d_4	d_3 d_5	d_3 d_6

(a) (3,4)门限方案

$d=d_1+d_2$
$d=d_3+d_4$

Server1	Server2	Server3	Server4
d_1 d_3	d_1 d_4	d_2 d_3	d_2 d_4

(b) (2,4)门限方案

图 14.2　门限方案示例

(1) 客户端任意选择 t 个 Share Server，把消息 M 发送给它们，同时要发送它选择的 t 个 Share Server 的编号。

(2) 收到签名请求的 Share Server，根据客户端选择的 Share Server 组合，选择相应的 d_i，计算 R_i(计算方法同基本方案)，发送回客户端。

(3) 客户端计算收到的 t 个 R_i 之积，得到最终签名 R(原理同基本方案)。

(4) 客户端使用 CA 公钥验证签名结果是否正确。

14.1.2　部分签名验证

ITTC 方案同时还提供了对部分签名验证的方法，以检查部分签名发生错误的 Share Server。这个方法需要客户端事先拥有一个正确的部分签名。

假设 Share Server i 以及给客户端发回了一个部分签名 $R_i=M^{d_i} \bmod N$，并且客户端在事先已经拥有了一个正确的部分签名 $V_i=g^{d_i} \bmod N$，其中 g 是一个事先选择好的随机数。

验证部分签名的步骤如下。

(1) 客户端随机选取 a 和 b，满足 $1\leqslant a,b\leqslant N$，并计算 $Z=M^a g^b \bmod N$。

(2) 客户端发送 Z 给 Share Server i，要求其对 Z 进行部分签名，并对结果利用 SHA-1 进行杂凑。

(3) Share Server i 计算 $A_i=\text{SHA-1}(Z^{d_i} \bmod N)$，并将 A_i 返回给客户端。

(4) 显然，如果 Share Server i 的部分签名一直都是正确的，则必然有

$$Z^{d_i} = (M^a g^b)^{d_i} = M^{d_i a} g^{d_i b} = R_i^a V_i^b \bmod N$$

从而 $A_i = \text{SHA-1}(R_i^a V_i^b \bmod N)$。客户端只需验证此式是否成立，如果不成立，则 Share Server i 的签名必然有错。

这个部分签名验证协议是文献[5]中的协议的简化版，ITTC 说明了一个发生错误的 Share Server 也有可能通过上述协议的验证，成功欺骗客户端，但是这种情况发生的概率不超过 $1/N^{1/2}$。对一个 1024b 的 RSA 算法而言，欺骗成功的概率不超过 $1/2^{512}$。这个概率已经是非常的小，在实际使用中已经足够证明部分签名的正确性。

14.1.3 优、缺点分析

该协议具有简单的结构和很好的安全性等优点，其密钥产生也采用了分布式方案，没有集中的处理。但存在以下一些不足之处。

(1) 部分密钥的分发和管理比较困难。增加一个服务器时，必须对每一个在线的服务器分配密钥数据；同时客户端也必须知道这样的增加，这对客户端提出了较高的要求。

(2) 当服务器很多时，服务器的密钥存储会迅速增加。设服务器的数量为 k，则每个服务器存储密钥的数量至少为 C_k^{t-1} 个。其中 C 表示组合，当 $k=10$，$t=3$ 时，$C_{10}^2=45$。

(3) 存在必须事先同步的问题。在进行计算前，必须先选定 t 个服务器。当 t 个服务器选择完成后，必须找到与这 t 个服务器匹配的数据组并通知他们。当其中有一个服务器被破坏时必须重复从选择服务器开始的整个过程。

14.2 基于 Shamir 秘密共享协议的入侵容忍协议

在 Shamir 秘密共享协议中，任意取 t 个部分密钥就能够重构秘密密钥。通过这种思路设计不重构但能够进行签名的系统是容易想到的。由于 Shamir 秘密共享协议中必须先恢复秘密密钥，而入侵容忍系统在任何情况下都不恢复秘密密钥。我们知道，在 Shamir 秘密共享协议中，设 $f(x) = \sum_{i=0}^{t-1} a_i x_i$，则利用拉格朗日插值公式，有

$$f(x) = \sum_{i=1}^{t}\left(f(x_i)\prod_{j=1,\cdots,t}^{j\neq i}\frac{x-x_j}{x_i-x_j}\right) \tag{14-1}$$

任选 t 对 $(x_i, f(x_i))$，就可以得到

$$a_0 = f(0) = \sum_{i=1}^{t}\left(f(x_i)\prod_{j=1,\cdots,t}^{j\neq i}\frac{-x_j}{x_i-x_j}\right) \tag{14-2}$$

可以设置 a_0 为秘密密钥 d。此时，对一个 Hash 值 M 的签名可通过公式(14-3)计算，即

$$M^d = M^{f(0)} = \prod_{i=1}^{t} M^{f(x_i)\prod_{j=1,t}^{j\neq i}\frac{-x_j}{x_i-x_j}} = \prod_{i=1}^{t} M^{b_i} \tag{14-3}$$

其中

$$b_i = f(x_i)\cdot c_i = f(x_i)\prod_{j=1,t}^{j\neq i}\frac{-x_j}{x_i-x_j} \tag{14-4}$$

这样可以将秘密 d 分到 k 个服务器中去($k \geqslant t$)。每个服务器计算 M^{b_i}，然后由一个合成器将结果乘起来就得到 M^d 而任何服务器都不会泄露秘密 d。

其难点也是明显的。由于计算 b_i 的过程中有除法计算，这样就会产生有理数(分数)，这使得升幂的过程包含了一个开方的过程。如 $M^{8/3}$ 就要求开 3 次方。解决这个问题的简单方法就是寻找一个数 v 构作一个域或环 Z_v，让所有的计算在这个域或环中进行。其中，v 和 x_i 的选取必须满足下列条件。

条件 14.1 v 为素数，或者保证所有 x_i 构成的 t 阶 Vandermonde 矩阵的行列式的值与 v 互素。

即使选取了满足条件 14.1 的 v 和 x_i，由于 b_i 的计算是对 v 求模，所以实际的结果是可能减掉或增加了一些 v 的倍数。最后的计算结果就可能是：$\prod_{i=1}^{t} M^{b_i} = M^{d+wv}$。

由于 $M^{\Phi(N)} \equiv 1 \bmod N$，所以，许多人想到了让 $v=\Phi(N)$。但选择 $v=\Phi(N)$ 使 x_i 选取大大受限于条件 14.1。并且，如果选择 $v=\Phi(N)$，则分享服务器就必须知道求一些特别数字对于 $\Phi(N)$ 的逆(将分母变到分子上去)，然而知道了某元素 c 与其对 $\Phi(N)$ 的逆 c^{-1}，就可以求出 $\Phi(N)$。而求出 $\Phi(N)$ 也就求出了私有密钥。显然这是一种不安全的入侵容忍方案。

CertCom 公司的 Frankel[5] 等人提出让多项式的系数 a_i 在 $\{0, L, \cdots, 2L^3N^{2+\epsilon}t\}$ 中，其中 $L=k!$。并取 x_i 属于 $[1,2,\cdots,k-1]$。由于所有 $f(x_i)$ 都能够整除以 L，故 b_i 的计算去掉了求逆的操作，可以在整数中进行。该方案可以对一般的 RSA 算法进行而不需要 RSA 的模是强素数。由于其参数的选取大大受限，带来了算法原理及安全性证明的复杂性。当 b_i 由 Share Server 进行计算时，也存在着如 ITTC 方案中的同步问题。因为 b_i 依赖于 x_i 的选取。Frankel 等人也对单个 Share Server 的验证提供了解决方案。

从其数学方案的描述中可以看出，该方案的特点和存在的不足如下。

(1) 采用平等的秘密分享，即分享的单层式结构。

(2) 参数的选取是受限的，限制了其使用，同时证明其安全性和编程保证十分复杂，增加了出现漏洞的可能性。

(3) 存在着需要同步的问题，而如果去掉同步，则会大大增加合成器的计算量。

Shoup[6] 提出了一个使用 RSA 强素数的方案，其保密的素数 $p=2p+1, q=2q+1$。所有的插值方程可以看成在模 $m=pq$ 的环中进行，但 Share Server 并不知道 m。因为 M^{4m} 模 N 等于 1，分开计算时增加一次平方，Combiner 再分别对各个结果平方一次从而得到 $M^{4\Delta(gm+d)}$，其中 $\Delta=(k!)^2$ 而 g 为整数。由于 $M^{4\Delta gm}=1$ 从而消除了 m 的影响。该方案中 c_i 是由 Combiner 完成的，消除了计算前的同步问题，但带来了 Combiner 的计算难度，使 Combiner 的计算性能下降。该方案中，Combiner 必须计算

$$W = y_{i_1}^{2\lambda c_{i_1}} y_{i_2}^{2\lambda c_{i_2}} \cdots y_{i_t}^{2\lambda c_{i_t}}$$

其中 y_i 为各 Share Server 的计算结果，$\lambda=k!$。显然，当 k 较大时，Combiner 的计算量比较大。

可以看出，该方案的特点和存在的不足如下。

(1) 是一种由各分享服务器组成的单层式分享结构，其合成器不存储任何秘密，可以由任何设备完成合成工作。

(2) 最后结果 $M^{4\Delta(gm+d)}$ 中消除了 g 的不确定性带来的影响。

(3) 消除了预先同步。

(4) 仅从理论上说明当增加或删除分享服务器时不会影响其他设备,但只提供了数学公式,没有任何实现上和系统结构上的说明。

(5) 要求使用强素数,会对某些应用带来限制。

(6) 整个计算当 k 较大时,合成器(客户端)的计算量比较大。

14.3 Jing-Feng 入侵容忍 CA 协议

文献[7]针对 ITTC 方案的一些缺陷提出了一个新的入侵容忍 CA 协议,称为 Jing-Feng 入侵容忍 CA 协议。随后又进一步发展了相关理论[8,9]。

14.3.1 系统结构

Jing-Feng 方案以 RSA 算法为基础,其基本原理类似于文献[1]中的思想。但 Jing-Feng 方案的结构不同于 ITTC 方案的结构,它消除了预先同步的问题,添加了系统连接的层次,减少了管理开销。图 14.3 给出了 Jing-Feng 方案的结构原理。

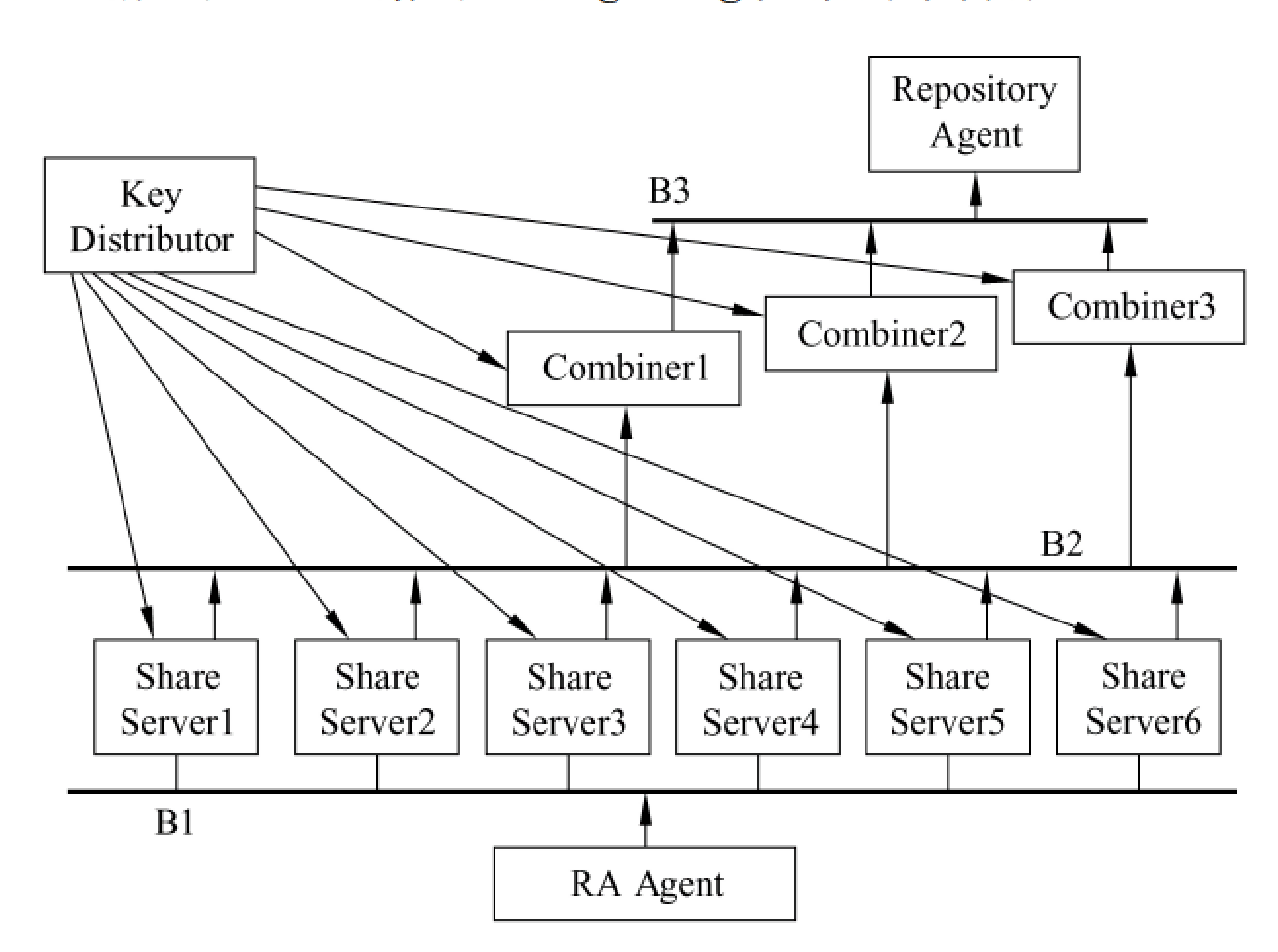


图 14.3 Jing-Feng 入侵容忍 CA 协议系统结构

Jing-Feng 入侵容忍 CA 协议主要包括以下 5 个部件。

(1) RA Agent:系统与 RA 的接口,它负责与 RA 进行保密通信,并检查 RA 的签名。同时,它也是通向外部网络的一个途径。在用户直接申请证书时,也通过此接口。

(2) Share Server:这些服务器是用于 CA 签名的,每一个都掌握着一个 CA 私钥的部分信息。每一个服务器都有自己的编号。RA Agent 通过广播信道 B1 与这些服务器相连。当有证书 cer 需要 CA 签名时,RA 为此签名设定一个可以区分的任务号 Task(cer)。RA Agent 通过广播将需要签名的信息与设定的任务号 Task(cer)广播到 Share Server 的网络上。Share Server 根据设定的算法选择任务,使用手中持有的部分私钥进行签名。

(3) Combiner:这些服务器是最后的计算单元,并且同样掌握着 CA 私钥的部分信息。当 Share Server 完成部分签名后,将把部分签名结果发送给 Combiner,Combiner 根据收到的部分签名信息和自己所拥有的部分私钥信息,进行最后的签名合成,形成最后的签名

结果。

(4) Key Distributor：即密钥分发中心，Key Distributor 作为一个可信的密钥分发中心，负责更换 Share Server 和 Combiner 的部分密钥。这个部件平时是离线的。由安全策略和管理规定与 Share Servers 的联网时间和方式，连接的方式可以采用经典的密钥注入的方式。

(5) Repository Agent：这个部件是与数据库的接口，同时负有系统错误检测和报警的任务。当 Combiner 计算验证错误时会通知 Repository Agent。Repository Agent 可以通过其他 Combiner 的计算结果查找发生问题的服务器或 Combiner。

14.3.2 密钥分发与签名

这里只考虑设计 k 个 Share Server 取 t 个得到签名的操作模式。CA 的私钥为 d，公钥为 e 和 N，当然 $\Phi(N)=(p-1)(q-1)$ 也是保密的。Share Server i 记为 S_i。

1. 密钥分发过程

Key Distributor(密钥分发器)随机选定 k 个小于 d/t 的随机数 d_i，该数可以远小于 d 但有效长度不应低于 200b。Key Distributor 根据针对每任意 t 的组合计算：

$$c_j = d - (d_{i_1} + d_{i_2} + d_{i_3} + \cdots + d_{i_t})$$

并保存 $i_1, i_2, i_3, \cdots, i_t$ 和对应的 c_j。对 k 个 d_i 来说共有 C_k^t 个组合，即计算 C_k^t 个 c_j。当 $k=10, t=3$ 时，共有 120 组。将这 C_k^t 组数$(i_1, i_2, i_3, \cdots, i_t, c_j)$分成几个部分，然后将每一部分送到不同的 Combiner。其中 $i_1, i_2, i_3, \cdots, i_t$为 Share Server 的 ID 号。

2. 签名协议

(1) RA Agent 分配任务号。每一个 RA Agent 都有自己的唯一标识 x，每一个 RA Agent 也维持一个序列号，当分发一个签名任务时，其任务序列号加 1。标识 x 加上自身的任务序列号就构成了任务号。这个任务号是全网唯一的，然后将签名任务广播到广播信道 B1。

(2) Share Server i 收到任务后，计算忙闲因子 F_i(Task(cer))。将待处理队列中的任务数乘以本机的性能指标，得到忙闲因子 F_i(Task(cer))。其中性能指标是在空闲状态下计算一次标准升幂的时间。

(3) 接收针对此任务的忙闲因子的广播，抛弃那些从同一个 IP 地址来的重复报文，并将接收到的忙闲因子按大小排序。如果机器空闲，将该任务送入待处理队列，任务会进入第(5)步执行。

(4) 如果存在 m 个服务器的忙闲因子小于自己的忙闲因子时，抛弃该任务；否则将该任务送入待处理队列，任务会进入第(5)步执行。

(5) 计算 $y_i=(\text{Hash}(\text{cer}))^{d_i}$。其中 Hash 为杂凑函数。Share Server 将计算的结果连同 cer、任务号 Task(cer)和自己的代号 i 送往广播信道 B2。

(6) Combiner 取用 t 个结果数据报，并寻找组合$(i_1, i_2, i_3, \cdots, i_t, c_j)$。如果找到，就计算

$$R = (\text{Hash}(\text{cer}))^{c_j} \cdot \prod_{i=i_1}^{i_t} y_i$$

R 就是应该得到的签名。如果没找到，重新取用另外 t 个结果数据报，重复第(6)步。穷举所有可能的结果组合，仍旧没有找到匹配的 c_j 则抛弃该任务。

(7) Combiner 利用公开密钥验证 R 的正确性，验证正确就将 cer、各服务器的 ID 号与 R 一起通过广播信道送往 Repository Agent；否则，向广播信道广播任务失败消息。其他 Combiner 收到任务失败消息后将启动计算，重新计算该任务，验证合格后都通过广播信道送往 Repository Agent。计算失败的 Combiner 重新选择另外不同的 t 个结果（如果有的话）重新计算。

(8) Repository Agent 将正确的 cer 与 R 一起送往数据库存档和被查，并通知其他 Combiner 停止该任务。Combiner 收到任务完成消息后抛弃该任务的数据。

14.3.3 多分享密钥方案

Combiner 可以安装在 Share Server 的后面，比 Share Server 更难受到攻击。但即使敌人全部掌握一个 Combiner，他也不能推出秘密密钥 d。即使一个 Combiner 拥有了全部的组合，也无法求解 d。可以证明，当 c_j 已知，d 与 d_i 未知时，所有具有 $c_j=d-(d_{i_1}+d_{i_2}+d_{i_3}+\cdots+d_{i_t})$ 这样形式的方程全部合在一起，其系数矩阵的秩为 k，而变量个数为 $k+1$。所以，直接通过一个 Combiner 无法求得秘密密钥。

虽然 Combiner 从广播信道上获取信息也不能推出任何 d_i，但是，当一个 Combiner 得到所有的组合后，Combiner 能够计算出任意的 d_i-d_j 的值。也就是说，他能够通过 d_i 求出 d_j。这样，一个 Combiner 就能够与一个 Share Server 合作求出秘密 d。这是人们所不希望的。

文献[7]通过将不同的组合分配给不同的 Combiner 来阻止 Combiner 与 Share Server 的合谋攻击，并提出一个 Combiner 的安全条件，即在同一个 Combiner 内，任何方程的线性组合得到的新的方程，其变量的个数大于 t。这样就保证了一个 Combiner 必须同 t 个 Share Server 合谋才能得到秘密密钥。从理论上讲，当每个 Combiner 只有一个方程时，该条件就得到了满足。这也说明 Jing-Feng 方案在理论上是可行的。

文献[7]进行了穷举分析，对 $k=5$，$t=3$ 来说，需要 8 个 Combiner 才能放进所有的组合并且保证满足 Combiner 的安全条件。因为需要的 Combiner 数量太多，这样的方案是不能采用的。文献[7]中设计了多分享密钥方案解决了这个问题，下面就来介绍这一方案。

由于 Share Server 的分享密钥较短，可以让每个 Share Server 拥有多个子密钥。只要 Share Server 不被破坏，则他一定可以计算出正确的多个结果。如 Share Server 1 拥有两个子密钥 d_{11} 和 d_{12}。在计算时，Share Server 1 求出两个结果 $y_{11}=(\text{Hash}(\text{cer}))^{d_{11}}$ 和 $y_{12}=(\text{Hash}(\text{cer}))^{d_{12}}$。将两个结果广播到网络上。

如果每个 Share Server 拥有 m 个子密钥，对于工作协议的准备就不再是简单的 C_k^t 个组合，而是 $C_k^t * t^m$ 个组合，即对应这么多组合可以求出 $C_k^t\times t^m$ 个不同的 c 值。

为减少 Combiner 的数量，将机器相同的组合定义为等价组合，于是得到了 C_k^t 个等价组合集。每个等价组合集中只要有一个组合在 Combiner 中，就在任何大于 t 台 Share Server 正常的情况下，能够求出正确的签名。设定了一个简化的安全条件，并在每个等价组合集中选取一个组合做代表，然后根据安全条件，进行 Combiner 的分配。下面来描述具体的操作过程。

离线的子密钥分发器从 $k\times m$ 个第一子密钥中取出 t 个，根据方程 $d=d_{1i_1}+d_{2i_2}+d_{3i_3}+\cdots+d_{ti_t}+c_a$ 再通过作减法可以求出 c_a，共有 $C_k^t\times i^m$ 种取法，故可以计算出 $C_k^t\times i^m$ 个方程的 c_a。这里，$n=C_k^t\times i^m$ 代表组合，表示有 n 个 c_a 值和 n 个方程组合表示，每一个方程组合表示是方程所对应的 t 个第一子密钥 d_{ji} 下标标号(或者说是变量的序号)的组合，这样的一个方程所对应的密钥 d_{ji} 的标号为密钥组合。显然，一个方程组合内的不同的第一子密钥是在不同的秘密分享运算器内，由于方程组合表示只与第一子密钥 d_{ji} 下标标号两位数字 ji 有关，所以，组合表示不泄露任何子密钥的信息。比如，每个方程组合表示含 t 个项，每个项的数字是第一子密钥 d_{ji} 的下标标号 ji，如(12,23,31)就表示 $t=3$ 时的一个方程组合表示。

根据上面定义的等价组合，可以知道(12,33,41)、(11,31,42)和(12,33,42)为 3 个等价组合。其中，只要存在任意一个组合，就表示只要这 3 台机器(Share Server)，即 1 号、3 号和 4 号机器正常时就能够求出签名结果。当然，Share Server(秘密分享计算器)计算的所有结果对最后的结论可能没有用处，这些结果可以在系统错误时进行冗余，并帮助找到错误的机器。

由于从 Combiner(秘密分享合成器)来看，其 c_a 是已知的，其他都是未知的变量。合成器安全条件是根据其方程和系统安全性要求提出的，安全条件是：一个秘密分享合成器内所含方程经线性组合得到的新方程，其变量的个数大于 t。

满足 Combiner 安全条件，就可以避免合成器与分享运算器对系统的合谋攻击。但是，该条件过于复杂，无法用程序或流程实现。因此为了实现安全的合成，必须有一个可行的算法。为实验目的，利用方程的特殊性，可使安全条件简化为：一个合成器内由任意两个方程的线性组合得到的方程，其变量的个数大于 t。

离线的子密钥分发器对所有的方程组合表示、等价组合集及对应的 c_a 值，按合成器简化安全条件通过穷搜进行分组，每一分组内的方程组合表示是有限的，再按分组的个数对应设置秘密分享合成器的个数，对应存储这些分组的方程组合表示及对应的 c_a 值。

只要从每个等价组合的集合中取出一个组合作代表，然后将所有等价组合集合的代表组合放到秘密分享合成器中，就可以针对每种机器组合得到正确的签名结果。

下面仍以 $t=3$、$K=5$ 为例，说明子密钥分发器计算并分发组合表示的步骤。

(1) 先忽略一个秘密分享运算器有多个第一子密钥的问题，根据 C_k^t 求解全部 10 种机器组合。

(1,2,3)、(1,2,4)、(1,2,5)、(1,3,4)、(1,3,5)、(1,4,5)、(2,3,4)、(2,3,5)、(2,4,5)、(3,4,5)，是 5 台秘密分享运算器任选 3 台的组合结果。

(2) 对上述求出的 10 种结果进行扩展，即对每个结果求出等价组合，形成 10 个等价组合集，当每个秘密分享运算器有两个机内密码号时，每个结果还有 $2t-1$ 个等价的组合，求出这些组合并将他们放在一个等价组合集里，即每个等价组合集中有 $2t$ 个等价的组合，可随机排序，如结果(1,2,3)的等价组合及随机排序为(11,21,31)、(12,21,31)、(11,22,31)、(12,22,31)、(11,21,32)、(12,21,32)、(11,22,32)、(12,22,32)，形成一个等价组合集。

(3) 将所有的等价组合集放在一个大组中，随机排序，并准备若干个分组。

(4) 从大组中依序取出一个等价组合集，从该等价组合集里依序找出一个组合，根据简化的安全条件，通过穷搜进行分组，即看这个组合能不能放到第一个分组里，如果能则将该

组合放入第一分组里，且一个等价组合集中只要有一个组合放入了一个分组中，就抛弃该等价组合集；如果不能，仍将其放回到该等价组合集里，并重新从该等价组合集中依序取出下一个组合，再根据简化的安全条件，看它能不能放到第一个分组里，如果所有该等价组合集里的组合都不能符合简化的安全条件，即不能放到第一个分组中，用同样的方法试第二个分组，如果还不行，试第三个分组，依次下去。该过程执行完成后，将所有不空分组的组合表示取出，每个分组中的组合表示通过管理允许的方式对应预存入一个秘密分享合成器中。

(5) 重复第(4)步，直到大组中所有的等价组合集都被取走到分组中为止。

(6) 统计有多少个分组中有组合，并将一个分组中的组合对应送入一个秘密分享合成器中预存。

(7) 在每个有组合的分组中，增加一些能满足安全条件的其他组合，以增加冗余。但在增加冗余的过程中，当一个等价组合集中有一个组合放入了一个分组中时，是不抛弃该等价组合集的。

通过上述步骤，对于 5 个秘密分享运算器的 2 和 3，且每个运算器存有两个第一子密钥的机内密码的情况，计算结果如下：

第一个分组内的 9 个组合如下。

(11,21,31)
(11,22,41)
(11,32,42)
(21,32,41)
(12,21,51)
(12,31,52)
(22,31,51)
(21,42,52)
(12,22,32)

第二个分组中的 12 个组合如下。

(11,41,51)
(31,41,52)
(11,21,42)
(12,31,42)
(21,32,41)
(11,22,52)
(12,32,51)
(21,31,51)
(22,42,51)
(32,42,52)
(12,22,41)
(12,21,52)

上述结果说明，只要有两个秘密分享合成器就能够对付 5 个秘密分享运算器。因为实际的等价集合只有 10 个，而现在两个秘密分享合成器中已经放置了共 21 个组合，说明已经

增加了很多冗余。

当选择 6 台秘密分享运算器且每个秘密分享运算器存两个第一子密钥的机内码时，第一个分组中可有 16 个组合，第二个分组中有 18 个组合。两个秘密分享合成器已经能够产生足够的冗余(由于组合数太多，在此不一一列举)。

实施时，可让 d_{ji} 的比特数远远小于(至少小于 4 倍)c_a 的比特数，如当 d 为 2048b 的数时，c_a 为 2048b 的数，d_{ji} 为 500b 的数或更少，以保证秘密分享运算器的运算速度，从而提高整个数字签名系统的运算速度。

每个秘密分享合成器并不存储针对所有子密钥 d_{ji} 的所有组合，但所有秘密分享合成器存储内容和的结果能保证包含针对秘密分享运算器的所有组合。

由于 Share Server 中密钥的长度远小于签名密钥的长度，所以增加子密钥基本不影响 Share Server 的计算性能。而对于 Combiner 来说，升幂计算只进行一次。由于方程个数并没有增加，故查找匹配的时间也没有太多的变化。

14.3.4 Jing-Feng 方案的安全性分析

首先，一个 Share Server 的泄露或被敌人掌握只能泄露其掌握的 d_i。因为 d_i 是随机选择的，单个 d_i 与 d 没有任何关系。所以一个 d_i 不暴露秘密密钥 d 的任何信息，即条件信息熵 $H(d|d_i)=H(d)$。由于 d_i 与 d_j 当 $i\neq j$ 时是独立选取的随机变量，所以有 $H(d|d_i,d_j)=H(d)$。即使多个随机的 d_i 也不反映 d 的任何信息。从 d_i 的产生过程来说，任意个 Share Server 的泄露都不泄露秘密密钥 d。

其次，通过 Share Server 到 Combiner 的广播信道也不能掌握秘密信息 d。在广播信道上，只有 $y_i=(\text{Hash}(\text{cer}))^{d_i}$ 能够反映秘密密钥，但通过 y_i 求 d_i 是一个离散对数问题，具有与 RSA 算法基本相同的困难性。也就是说，通过广播信道得不到关于任意一个 d_i 的信息。这是整个 CA 安全的基本保证。

在得不到任何 d_i 的情况下，任意多个 Combiner 合谋进行攻击也只能得到最多所有组合的方程。前面已经说明，所有方程合起来的秩只有 k 而变量的个数却是 $k+1$。所以无法找到秘密 d。当每个 Share Server 采用两个以上的子密钥时，变量的个数增加了 k 个，但方程的数量却可以不增加。这更加增大了 Combiner 合谋攻击的难度。

由于系统满足 Combiner 安全条件。一个 Combiner 与少于 t 台 Share Server 合谋不能对系统构成威胁。但多台 Combiner 与多个 Share Server 合谋可能会对系统形成威胁。在满足 Combiner 安全条件时，合谋的攻击也是有条件的。合谋攻击成功的条件与对 Combiner 组合的发放算法密切相关。合理的组合发放算法能够增加合谋攻击的难度。利用 Share Server 的多个子密钥和 Combiner 的安全条件，抵制多个 Combiner 和一个 Share Server 合谋攻击是容易的。从理论上讲，随着单个 Share Server 子密钥的增加，变量个数会成倍增加，而放在 Combiner 的方程的个数却可以不增加。这样，单个子密钥在 Combiner 中的方程中出现的次数就会很少，在要求极高安全的场合，让每个子密钥只在一个方程中出现一次。在这种极限情况下，所有的方程组合就满足 Combiner 的安全条件。也就是说，多个 Combiner 联合就相当于满足 Combiner 安全条件的一个 Combiner，此时的安全是有保障的。

14.3.5 Jing-Feng 方案的特色

(1) 双层结构。采用了两层式的结构,将现有方案中的单层秘密分享分解成双层,克服了文献[1]的方案中分享器增加导致的密钥管理困难,也避开了其他方案中的复杂理论问题。同时,双层方案也增加了对手攻击的难度,从而增加了安全性。由于两层系统使用不同的算法、不同的系统结构,子密钥的类型、密钥管理和存储也都不一样,攻击两层就更加困难。而我们的两层结构保证,即使一层全部遭到攻击和控制,攻击者也没有办法获得 CA 私钥。

(2) 原理简单。采用的原理非常简单,也容易证明方案的安全性。同时,方案对参数的选择没有任何限制,不要求是强素数。

(3) 无计算同步问题。双层式秘密分享的方案突破了已有方案的同步缺陷。也就是说,任务的管理者不需要知道谁会参与计算,从而减轻任务管理系统的复杂性,将密钥的管理和运算彻底地同任务管理者或客户机分开。这从另一个方面来说,也增加了整个系统的可管理性和安全性,实现了一种自制的结构。同时,不同步可以使得当有一台设备故障时,任务的管理者不需要了解,也不需要重新发布任务。

(4) 添加或删除设备简单。当添加一台 Share Server 时,只需要由 Key Center 为它生成几个随机数,并分配代号(Key Center 也可以在第一次初始化时就生成好多个这样的参数)。然后将这些参数装入 Share Server。此时该 Share Server 就可以上线运行。然后针对该 Share Server 生成各个 Combiner 的组合和密钥,并将这些密钥加入到 Combiner 中,这样就完成了一个 Share Server 的增加。Combiner 的增加就更加容易。计算一个新的安全组合集,然后安装进新增的 Combiner 中,这样系统的增加就完成了。删除一台 Share Server 设备可以不做任何工作,也可以清除 Combine 中相应机器代号的组合。删除 Combiner 不需要做任何工作。

(5) 效率高。与其他方案相比,Jing-Feng 方案总的效率较高。计算主要消耗在升幂上。对 4096b 的 RSA 来说,Jing-Feng 方案的 Share Server 只计算 200b 的升幂,故 3 个 Share Server 就计算了 600b 的升幂,加上冗余计算的部分,就是 1200b 的升幂。Combiner 计算 4096b 的升幂。总共计算 5296b 的升幂。将签名所需的升幂为一次签名计算量,Jing-Feng 方案的总计算量为 1.3 次签名。如果降低 Share Server 所拥有密钥的长度,总的签名计算量还会降低。与其他方案相比,由于其他方案的单层结构限制,总的计算量都是 t 次签名以上。Shoup 的方案消除了计算前的同步,当 Share Server 的数目 k 较大时,其计算量大于 t 次。

对于产生一次签名的时间来说,Jing-Feng 方案比其他方案略慢。两层结构需要串行操作,故需要 1.3 次的签名时间。当忽略掉为部分签名检测而进行的计算时间,则其他方案只需要一次签名的时间。

总之,Jing-Feng 方案是在现有入侵容忍 CA 方案的基础上,提出的一套新的入侵容忍 CA 方案,其特点是使用了非常简单的原理,安全性容易证明;采用了双层体系结构,解决了计算前的同步问题,并提高了安全性能,且方案的效率也比较高。表 14.1 给出了几个典型方案的一个比较。

表 14.1　几个典型方案的比较

方　案	Jing-Feng 方案	ITTC 方案	Shoup 方案	Frankel 方案
一次签名的总计算量	1	t	t	t
签名时间	1	1	1	1
计算前同步	不需要	需要	不需要	需要
系统安全层	2	1	1	1

14.4　自治协同的入侵容忍 CA 协议

CA 私钥的安全保护问题是 PKI 系统设计中需要考虑的首要问题，为了保护 CA 私钥，已经提出了很多种方案，包括 CA 离线方案、入侵容忍 CA 方案等。其中 Jing-Feng 入侵容忍 CA 方案是一个相对比较先进的方案，具有明显的优点，其结构简单、性能高效且安全性也比较高，其多级结构不仅有创新性，而且也提高了安全性。但是它还存在以下一些不足之处。

(1) 采用一个密钥分发中心来产生和分割 CA 私钥是危险的，一旦这个中心被攻击者控制，PKI 系统的核心秘密将受到危及。实际上，在现实中，寻找或建立一个完全可信的中心都是极其困难的。

(2) 如果 Share Server 或 Combiner 受攻击者控制后，给出了一个错误的签名，系统没有能力发现是哪一台服务器发生了错误，因为该方案还缺乏部分签名的验证。

(3) 由于无法发现是哪一台服务器的签名出错，在这种情况下，系统就被迫必须全面更新所有的部分私钥，尽管其中的一些更新是不必要的。

考虑到上述问题，在借鉴了 Jing-Feng 方案优点的基础上，在其系统结构上又进行了改进，以达到自治与协同的目标。

我们加强了每个服务器的独立性，使其不再依赖于外部一个可信的中心，从部分私钥的产生、部分私钥的更新到部分签名都由服务器自行完成，实现自治的目标。另外，整体的 CA 私钥和签名结果，是由所有的服务器运用分布式的算法协同产生的，没有任何一台服务器能单独决定或获得 CA 私钥，同样也不能单独完成签名，只有达到了门限值数量的服务器才能完成这一切，从而实现更好的入侵容忍效果。

该方案是一个完整的入侵容忍 CA 的分布式建立和安全运行方案。它具有以下 4 个主要特点。

(1) CA 的密钥和部分私钥都是由所有服务器分布式产生，不再由一个单独的密钥分发中心产生并分割 CA 私钥，因为这样一个中心将成为系统的最大弱点，并且违背了入侵容忍的基本思想。

(2) 在 CA 运行一段时间后，可以对所有服务器的部分私钥进行更新，而并不改变 CA 的私钥。

(3) 当有一台服务器给出了错误的部分签名结果时，系统可以迅速地找到是哪台服务器发生了错误。

(4) 系统在运行上仍然保持了 Jing-Feng 方案的性能。

为描述方便起见，本节采用以下记号和假设。

记 RSA 的全部参数为 $K=(N,P,Q,e,d)$，其中 N 和 e 是公钥，d 是私钥，P 和 Q 是构成 N 的两个大素数因子，即 $N=PQ$。欧拉函数 $\varphi(N)=(P-1)(Q-1)$。

假设共有 k 台 Share Server，编号为 $S_1 \sim S_k$，c 台 Combiner，编号为 $S_{k+1} \sim S_n (n=k+c)$。考虑 t 台 Share Server 加 1 台 Combiner 的签名模式。系统满足 $(t+1,n)$ 门限签名方案的要求。

在一个 (t,n) 门限签名方案中，将 CA 私钥 d 在所有服务器集(包括 Share Server 和 Combiner) P 中进行分配，n 表示服务器个数，t 是门限值 $(1 \leqslant t \leqslant n)$，满足以下条件。

(1) P 中的任意一个子集 A，只要其包含的元素大于等于 t，即 $|A| \geqslant t$，该子集就能联合恢复 CA 私钥 d。

(2) P 中的任意一个子集 A，只要其包含的元素小于 t，即 $|A| < t$，该子集就不可能联合恢复 CA 私钥 d。

(3) 满足条件(1)的子集就能完成一个 CA 签名，而满足条件(2)的子集无法完成或者伪造 CA 签名。

(4) 在 CA 私钥的整个生命周期内，包括 CA 私钥的产生、拆分、使用，该私钥都不会被重构出来。

(5) 签名过程不影响每个部分私钥的安全性。

14.4.1 系统结构

CA 协议系统结构如图 14.4 所示，除密钥分发中心外，系统的主要部件都继承自 Jing-Feng 方案，且仍然保持其双层签名结构。

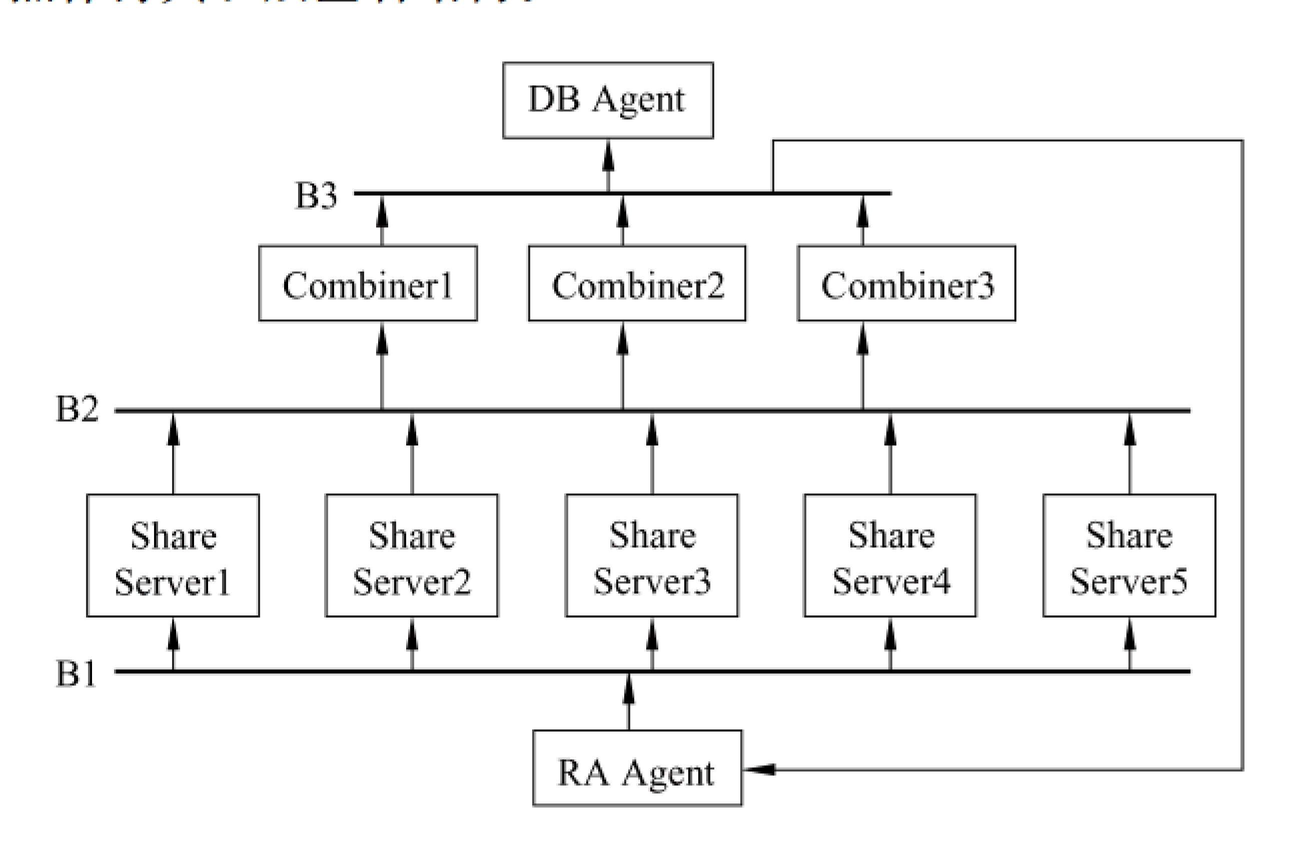


图 14.4 自治协同的入侵容忍 CA 协议系统结构

RA Agent 是与 RA 系统的接口，负责从 RA 系统接收签发证书的请求，然后将需要签名的消息发送给内部的 Share Server。Share Server 和 Combiner 是负责签名的 CA 部件，各自拥有自己的部分私钥信息。签名时，Share Server 使用它们拥有的部分私钥做部分签名，而 Combiner 根据 Share Server 的计算结果及拥有的部分私钥信息，合并计算出最终的签名结果。签名结果同时发送给 RA Agent 和 DB Agent。DB Agent 是与数据库的接口，

在获得 Combiner 的计算结果后，将其存入数据库中。

14.4.2　密钥分配

CA 私钥的拆分方式保持了 Jing-Feng 方案的特点，即每个 Share Server $S_i(1\leqslant i\leqslant k)$ 掌握着一个随机数 d_i 作为部分私钥。然后针对这 k 个随机数中的任意 t 个数的组合，计算

$$c_j = d - (d_{i_1} + d_{i_2} + \cdots + d_{i_t}) \quad (14\text{-}5)$$

并保存 $i_1, i_2, \cdots, i_t$ 和对应的 c_j，总共产生 C_k^t 个组合。最后将这 C_k^t 个$(i_1, i_2, \cdots, i_t, c_j)$平均分配给所有的 Combiner 保存。此外，为了部分签名的更新和验证，每台服务器还需要保存一些额外的信息，这将在下面介绍。

14.4.3　CA 密钥的产生与分割

这里假定 CA 系统使用的是 RSA 算法，其 RSA 密钥的产生可分为以下两步。

(1) 由 Share Server 和 Combiner 合作分布式产生两个大素数 P 和 Q，然后分布式计算 $N=PQ$，在这一过程中 P 和 Q 不会被任何一台服务器获知，最终大家都只知道 N 的值。

(2) 由于 RSA 的公钥指数 e 是一个公开的值，并且通常是一个定值，所以只要根据 N 和 e 的值，分布式决定私钥 d，产生各个服务器的部分私钥。同样在这一个过程中，没有一台服务器会得到 d 的具体值，它们唯一能得到的只能是自己拥有的部分私钥。

协议的前提假设是：在 CA 密钥的产生和分割过程中，都假定所有服务器是诚实的，即所有服务器都会如实地遵照协议要求进行运算和操作，但是协议保证了无论服务器是否诚实，都不会对 CA 的密钥安全产生威胁。另外，除广播以外，服务器之间的通信都是通过加密信道(如 SSL)进行的。为了加强安全，密钥产生和分割的整个过程中，CA 系统都处于离线状态。

14.4.4　公钥的分布式产生方案

这里介绍两种公钥的分布式产生方案。

1. 方案 Ⅰ

作为产生 CA 密钥的第一步，先产生 RSA 密钥中的 N。所有的 n 台签名服务器(包括 Share Server 和 Combiner)都共同参与这一过程，分布式产生 N。

文献[10]和[11]描述了一个在多台服务器间分布式产生 N 的基本协议，其基本过程都是源于一个相同的安全多方计算协议——BGW 多方计算协议，在此将同样采用一个简化的 BGW 协议完成 N 的计算。产生的基本过程如下。

(1) 每台服务器 $S_i(i=1,2,\cdots,n)$都产生两个随机大整数 p_i 和 q_i，具体位数取决于系统对 RSA 位数的要求。在下面的整个计算过程中，这些随机数都只有产生它的服务器知道，不对任何人公开。

(2) 采用简化的 BGW 协议分布式计算 $N=(p_1+p_2+\cdots+p_n)(q_1+q_2+\cdots+q_n)$。协议保证了在这一过程中，除了最终公开的 N 以外，不会有任何有关 p_i 和 q_i 的信息被泄露。

(3) 测试 N 是否是两个大素数的乘积，如果是，则过程结束；否则，重新返回第(1)步。

从这个基本过程可以看出，在计算结束之后，N 应该是两个大素数 P 和 Q 的乘积，且满

足 $P=p_1+p_2+\cdots+p_n$，$Q=q_1+q_2+\cdots+q_n$。所以，这个 N 是符合 RSA 算法要求的。

首先介绍简化的 BGW 协议。在此之前，每台服务器都已经产生了两个秘密的 p_i 和 q_i，这一步的工作就是计算 $N=(p_1+p_2+\cdots+p_n)(q_1+q_2+\cdots+q_n)$，且不泄露所有的 p_i 和 q_i 的任何信息。简化的 BGW 协议的基本过程如下。

(1) 首先系统必须事先约定一个大素数 P_0，满足 $P_0>N$，以下 BGW 协议的各项计算都是在模 P_0 下进行的。

(2) 设 $m=\left\lfloor\frac{n-1}{2}\right\rfloor$，每一台服务器 $S_i(i=1,2,\cdots,n)$都产生两个 m 次随机多项式 f_i 和 g_i 以及一个 $2m$ 次随机多项式 h_i，满足 $f_i(0)=p_i$，$g_i(0)=q_i$，$h_i(0)=0$。多项式的其余系数都是 Z_{P_0} 上的随机数。

(3) 每一台服务器 S_i 都分别计算下列 $3n$ 个值：

$$p_{i,j}=f_i(j),q_{i,j}=g_i(j),h_{i,j}=h_i(j)\quad j=1,2,\cdots,n$$

(4) 每一台服务器 S_i 都发送给服务器 S_j 一个三元组$(p_{i,j},q_{i,j},h_{i,j})(j=1,2,\cdots,n)$。这样，每一台服务器 S_j 就获得了 $p_{1,j},p_{2,j},\cdots,p_{n,j};q_{1,j},q_{2,j},\cdots,q_{n,j};h_{1,j},h_{2,j},\cdots,h_{n,j}$。

(5) 根据第(4)步获得的信息，每一台服务器 S_i 都能计算：

$$N_i=((\sum_{j=1}^{n}p_{j,i})(\sum_{j=1}^{n}q_{j,i})+\sum_{j=1}^{n}h_{j,i})\bmod P_0$$

然后广播 N_i，告知其他所有的服务器。

(6) 此时每一台服务器都获得了 $N_1,N_2,\cdots,N_n$。设 $N(x)$是以下多项式：

$$N(x)=((\sum_{j=1}^{n}f_j(x))(\sum_{j=1}^{n}g_j(x))+\sum_{j=1}^{n}h_j(x))\bmod P_0$$

显然有 $N(i)=N_i$，共 n 个方程。而 $N(x)$为 $2m$ 次多项式且 $2m<n$，所以每台服务器都可以根据此 n 个方程解出多项式 $N(x)$的各项系数，得到 $N(x)$的具体表达式。

(7) 最后，每一台服务器都计算 $N(0)$的值，而

$$N(0)=(\sum_{j=1}^{n}f_j(0))(\sum_{j=1}^{n}g_j(0))+\sum_{j=1}^{n}h_j(0)=(\sum_{j=1}^{n}p_j)(\sum_{j=1}^{n}q_j)=N$$

这样，就通过分布式多方计算获得了 RSA 密钥中 N 的值。文献[11]说明了在 BGW 协议中，即使有 m 台服务器共谋，也无法获得任何其他 p_i 和 q_i 的信息，因此也就无法得到 N 的分解信息。

接下来测试 N 的合法性。从上面的步骤可以看出，各个服务器的 p_i 和 q_i 都是随机选取的，因此计算出来的 N 可能并不符合 RSA 算法的要求，即 N 不一定是两个素数之积，为此必须检查 N 的合法性。当然，在这个测试过程中，仍然需要保证各自的 p_i 和 q_i 的机密性，绝不能泄露给他人。

首先要说明一个事实：设 $P=p_1+p_2+\cdots+p_n$，$Q=q_1+q_2+\cdots+q_n$，且都是素数，$N=PQ$，则对任意的 $a\in Z_N^*$，都有下式成立，即

$$a^{N+1-P-Q}\equiv 1(\bmod N)$$

这是因为由数论知识可知，对任意的 $a\in Z_N^*$，都有

$$a^{\varphi(N)}\equiv 1(\bmod N)$$

而

$$\varphi(N) = (P-1)(Q-1) = PQ+1-P-Q = N+1-P-Q$$

所以上述事实成立。

根据上述事实可以构造下面的测试协议。

(1) 首先在协议执行之前，系统必须事先约定一个素数 a。

(2) 服务器 S_1 计算 $a_1 = a^{N+1-p_1-q_1} \bmod N$，其余的服务器 S_i 计算 $a_i = a^{-p_i-q_i} \bmod N$。

(3) 所有的服务器都广播它们计算出来的 a_i。

(4) 每台服务器都可以计算下式，即

$$A = \prod_{i=1}^{n} a_i \bmod N$$

(5) 每台服务器都可以检查 A 的值，如果 $A \neq 1$ 则 N 不合法；否则测试通过。

首先该测试协议是可靠的(也称正确的或完备的、完全的)。这是因为

$$A = \prod_{i=1}^{n} a_i = a^{N+1-p_1-q_1-p_2-q_2-\cdots-p_n-q_n} = a^{N+1-(p_1+p_2+\cdots+p_n)-(q_1+q_2+\cdots+q_n)}$$
$$= a^{N+1-P-Q} \bmod N$$

根据上述事实，如果 P 和 Q 是素数，$A=1$ 就必然成立。所以逆否命题也成立，即如果 $A \neq 1$，可以判定 N 不合法。但是，$A=1$ 仅仅是 N 合法的一个必要条件而非充分条件，即如果 $A=1$，并不能判定 N 合法。所以有可能产生一个 N，它并不是两个素数之积，却又能通过上面这个测试协议。然而，文献[12]证明了这种情况出现的概率小于 $1/10^{40}$。因此，根据 $A \neq 1$，可以判定 N 肯定不合法；而 $A=1$ 的情况下，N 不合法的概率小于 $1/10^{40}$，从而能以压倒性的概率判定 N 是合法的。所以测试方案是足够可靠的。

其次，说明该测试协议是安全的。这是因为在整个方案中，每个服务器向外界泄露的只有 a_i。要想从 a_i 获得 p_i 和 q_i，实际上与攻击 RSA 算法相同，安全性仅仅取决于 p_i+q_i 的大小。所以，只要 p_i 和 q_i 足够大，在 RSA 算法是安全的假设下，测试协议也是安全的。

2. *方案Ⅱ*

方案Ⅱ在基本步骤上和方案Ⅰ相同，也分为 3 步：随机生成 p_i 和 q_i，分布式计算 N，分布式测试 N 的合法性。其主要区别在于协议的主体部分，即分布式计算 N 上。

关于多方计算 RSA 公钥 N 已有一些研究，方案Ⅰ中使用了 BGW 协议计算方法，在这个协议中，如果参与计算的服务器中有$\lfloor n/2 \rfloor+1$ 台共谋就可以得到 N。而文献[13]中也提出了一种多方计算协议，尽管这个协议的效率比文献[10]中的效率略低，但是保证了只有所有服务器共谋才可能获得 N，更大程度地保障了安全性。虽然已经有一些针对文献[13]中的两方协议的攻击结果，但却未能攻击其多方协议。因此，可以在文献[13]中的方法的基础上完成 N 的多方计算。产生 N 的具体协议如下。

(1) 每台服务器 $S_i(i=1,2,\cdots,n)$ 都产生两个随机正整数 p_i 和 q_i。在下面的整个计算过程中，这些随机数都只有产生它的服务器知道，不对任何人公开。以下的步骤就是计算 $N=(p_1+p_2+\cdots+p_n)(q_1+q_2+\cdots+q_n)$。协议保证了在这一过程中，除了最终公开的 N 以外，不会有任何有关 p_i 和 q_i 的信息被泄露。

(2) 每台服务器 $S_i(i=1,2,\cdots,n)$ 都随机产生一个 RSA 密钥对：公钥 (M_i, e_i) 和私钥 d_i。其中 M_i 应足够大，且满足 $M_i \geqslant 2N_L$(其中 N_L 是 N 的最大可能值，由系统事先选定)。

(3) 每台 S_i 再随机产生一组 $a_{ij}(j=1,2,\cdots,n)$，满足当 $i\neq j$ 时，$0<a_{ij}<N_L$，且 $\sum_j a_{ij}=0$。

(4) 每台 S_i 都广播以下四元组：

$$(M_i, e_i, p_i^{e_i} \bmod M_i, q_i^{e_i} \bmod M_i)$$

从而所有服务器都可以得到 n 个这样的四元组。

(5) 任取一对(S_i,S_j)，进行以下操作：S_i 随机产生 T 个 x_u(T 是一个预先约定的值)，满足$\sum_u x_u=1$，然后使用这 T 个随机数将$(p_iq_j)^{e_j} \bmod M_j$ 拆成 T 份，即$(p_iq_jx_u)^{e_j} \bmod M_j$ ($u=1,2,\cdots,T$)。运用相同的方法，S_i 可以将$(p_jq_i)^{e_j} \bmod M_j$ 和 $a_{ij}^{e_j} \bmod M_j$ 也同样拆成 T 份。这样S_i 就得到了 $3T$ 个数，然后将这 $3T$ 个数以随机顺序发送给 S_j。至此，S_j 可以使用其私钥 d_j 将这 $3T$ 个数分别解密，并将它们相加，得到

$$\begin{aligned} A_{ji} &= \sum_u p_iq_jx_u + \sum_u p_jq_ix'_u + \sum_u a_{ij}x''_u \\ &= (p_iq_j + p_jq_i + a_{ij}) \bmod M_j \end{aligned}$$

因为 M_j 是一个足够大的数，显然就有 $A_{ji}=p_iq_j+p_jq_i+a_{ij}$。

(6) 所有服务器两两之间都进行第(5)步的操作，完成后每台服务器 S_i 都可以得到 n 个 $A_{ij}(j=1,2,\cdots,n)$。此时 S_i 就可以计算 $N_i=\sum_j A_{ij}$。

(7) 每台 S_i 都广播其得到的 N_i。

(8) 所有服务器都可以计算

$$\sum_i N_i = \sum_i \sum_j A_{ij} = \left[\sum_i \sum_j (p_iq_j + p_jq_i)\right] + \sum_i \sum_j a_{ij}$$

由于 $\sum_j a_{ij}=0$，所以由上式可得

$$\sum_i N_i = 2(p_1+p_2+\cdots+p_n)(q_1+q_2+\cdots+q_n) = 2N$$

(9) 最后将第(8)步的 $\sum_i N_i$ 除以 2，就得到了 RSA 公钥 N。并且每台服务器都可以计算得到。

至此，所有服务器就通过分布式多方计算获得了一个 N 的具体值。文献[13]中指出，只有所有服务器联合，才可能获得破解协议，获取 CA 秘密，这显然已远远超出门限签名方案的要求。

方案Ⅱ的分布式产生 N 的协议，虽然与方案Ⅰ大不相同，但是它们最终获得的结果都一样，即每个 S_i 拥有 p_i 和 q_i，$N=(p_1+p_2+\cdots+p_n)(q_1+q_2+\cdots+q_n)$。所以，仍然采用方案Ⅰ的测试协议可以测试方案Ⅱ的 N 的合法性。

14.4.5 私钥的分布式产生方案

根据入侵容忍思想的要求，CA 私钥的产生应受到最严格的控制，这个过程同样也需要分布式完成，以保证在这个过程中没有一个服务器可以单独获知 CA 私钥，门限数量以下的服务器的联合，也无法从这个过程中窃取 CA 私钥。

为了达到保护 CA 私钥的目的，并不直接生成 CA 私钥 d，而是分布式生成每台服务器的部分私钥，而 d 实际上就是多个部分私钥的和。在这一生成过程中，每台服务器都无法获知私钥 d 的任何信息，也不对外泄露其部分私钥的任何信息。

通过 14.4.4 小节的计算，已经获得 RSA 密钥中的 N，RSA 公钥 e 也在事先约定，事实

上，实际应用中 e 通常是一个定值，如 $e=65537$。这一节就要根据 N 和 e 及各台服务器掌握的 p_i 和 q_i，通过分布式多方计算得出各个部分私钥。最终部分私钥的分割结果应是 14.4.2 小节中的分割结果：Share Server 各自拥有部分私钥 d_i，再根据式(14-5)计算出 C_k^t 个 c_j，将它们平均分配给 Combiner，分配方案遵循 Jing-Feng 方案的说明。

总之，我们的目的如下。

(1) 根据 14.4.2 小节的密钥分配方式，产生各个服务器的部分私钥，并且所有服务器拥有的部分私钥只有自己知道，在部分私钥的产生过程中，绝不能泄露给任何人。

(2) 产生的部分私钥，确实是 CA 私钥 d 的分割，这个 d 值由 N 和 e 确定，但所有人只知道这个 d 确实存在，却不能有任何人知道它确切的值。

(3) 只有门限数量的服务器联合，才可能获得确切的 d。

下面介绍两种方案。

1. 方案 Ⅰ

方案Ⅰ部分借鉴了文献[14]中的思想，但是文献[14]中的方案是针对所有服务器共同参与签名的 CA 结构，适用的环境与该方案大相径庭，无法在此使用。因此这里提出了一种适用于 Jing-Feng CA 系统的新方案。具体的分布式产生协议如下。

(1) 每台 Share Server S_i 都产生一个随机大整数 d_i，这个 d_i 就作为他们的部分私钥。

(2) 任取 t 台 Share Server(设为 $S_{i_1},S_{i_2},\cdots,S_{i_t}$)和一台 Combiner(设为 S_j)为一种组合，开始以下步骤。

(3) 所有的 n 台 S_i 都产生一个随机大整数 r_i。

(4) S_1 计算 $\varphi_1=N+1-p_1-q_1$，其他 S_i 计算 $\varphi_i=-p_i-q_i(1<i\leqslant n)$，则显然有 $\varphi(N)=\varphi_1+\varphi_2+\cdots+\varphi_n$。

(5) 每台服务器都各自计算 $\gamma_i=\varphi_i+r_ie$，然后广播 γ_i。

(6) 获得所有广播的 γ_i 后，每台服务器都计算下式，即

$$\gamma=\gamma_1+\gamma_2+\cdots+\gamma_n=\varphi(N)+Re$$

其中，$R=r_1+r_2+\cdots+r_n$。

(7) 由于 RSA 算法要求 e 必须与 $\varphi(N)$ 互素，显然 e 和 γ 也互素，从而所有服务器都可以根据扩展的欧几里德算法计算出 x 和 y，满足 $x\gamma+ye=1$，即 $x\varphi(N)+(xR+y)e=1$。令 $d=xR+y$，则 $ed=1-x\varphi(N)$，从而显然有 $ed=1 \bmod \varphi(N)$，据此可以用这个 d 作为 CA 私钥，并且此时也无人知道 d 的值。

(8) 每台服务器 S_i 按以下规则计算一个 K_i。

① 如果该服务器是第(2)步中选择的 Share Server，则其 $K_i=xr_i-d_i$。

② 如果该服务器是第(2)步中选择的 Combiner，则其 $K_i=xr_i+y$。

③ 如果该服务器不是以上两种情况，则其 $K_i=xr_i$。

(9) 第(2)步中选择的 Combiner S_j 产生一个随机大整数 E_0，然后将 E_0 发送给 S_1，S_1 计算 $E_1=E_0+K_1$，然后 S_1 把 E_1 再发送给 S_2，S_2 同样计算 $E_2=E_1+K_2$，…，以此类推，S_n 最后将得到

$$E_n=E_0+\sum_{i=1}^{n}K_i$$

(10) S_n 把得到的 E_n 再发送给 S_j，S_j 就可以计算

$$c_j = E_n - E_0 = \sum_{i=1}^{n} K_i = xR + y - (d_{i_1} + d_{i_2} + \cdots + d_{i_t})$$
$$= d - (d_{i_1} + d_{i_2} + \cdots + d_{i_t})$$

这样 S_j 就获得了 14.4.2 小节中说明的部分私钥分配结果。

(11) 反复执行第(2)～(10)步，每次都选取不同的 t 个 Share Server 组合和对应的 Combiner，生成不同的随机数 r_i。组合方案仍然使用 Jing-Feng 方案中说明的组合方法，共需反复执行 C_k^t 次。

至此，就完成了 CA 私钥的产生和分割过程，所有部分私钥的分配结果仍然保持了 Jing-Feng 方案的结构，因此，也保持了其入侵容忍方案的安全性和可靠性。

2. 方案Ⅱ

在方案Ⅰ中，每选取一个组合，都必须循环(2)～(10)步一次，为了提高效率，提出了新的方案Ⅱ，以减少循环中的计算量。具体的分布式产生协议如下。

(1) 服务器 S_1 计算 $\varphi_1 = N + 1 - p_1 - q_1$，其他 S_i 计算 $\varphi_i = -p_i - q_i (1 < i \leqslant n)$，显然有 $\varphi(N) = \varphi_1 + \varphi_2 + \cdots + \varphi_n$。

(2) 每台 S_i 随机产生 n 个 $\gamma_{ij} (j = 1, 2, \cdots, n)$，并且满足 $\varphi_i = \sum_j \gamma_{ij} \bmod e$。然后将 γ_{ij} 发送给 S_j，这样每台 S_j 都又获知了 n 个数 $\gamma_{1j}, \gamma_{2j}, \cdots, \gamma_{nj}$。

(3) 每台 S_i 都可以计算

$$\alpha_i = \sum_j \gamma_{ji} \bmod e$$

然后广播得到的 α_i。

(4) 根据(1)～(3)的结果，因为 $\varphi(N) = \varphi_1 + \varphi_2 + \cdots + \varphi_n$，$\varphi_i = \sum_j \gamma_{ij} \bmod e$，且 $\alpha_i = \sum_j \gamma_{ji} \bmod e$，显然有

$$\varphi(N) = \sum_i \alpha_i \bmod e$$

从而所有人在得到所有 α_i 后都可以计算一个 β，满足

$$\beta \sum_i \alpha_i = \beta\varphi(N) = -1 \bmod e$$

因为 RSA 算法要求 e 必须与 $\varphi(N)$ 互素，所以这个 β 必然存在。而且由于这是在模 e 下的计算，$\varphi(N)$ 的值仍然是无人知晓。根据这个结果，必然存在一个整数 d，满足 $\beta\varphi(N) = ed - 1$，即有 $ed = 1 + \beta\varphi(N) = 1 \bmod \varphi(N)$，这正好满足 RSA 算法要求，正是所要寻找的 CA 私钥 d。然而由于 $\varphi(N)$ 的值无人知晓，也就无人可以根据此式获知 d 的值，即

$$d = (1 + \beta\varphi(N))/e$$

(5) 现在，每台 S_i 都可以计算一个 d_i，即

$$d_i = \lfloor \beta\varphi_i / e \rfloor$$

显然有

$$d = r + \sum_i d_i$$

其中 r 是由于取整引起的偏差，并显然满足 $0 \leqslant r \leqslant n$。

(6) 为了找出这个偏差 r，任取一个消息 M，所有服务器都用自己的 d_i 计算 $M^{d_i} \bmod N$

并广播，从而大家都可以计算

$$R' = \prod_i M^{d_i} = M^{\sum_i d_i} \bmod N$$

然后针对 0～n 之间的任意 r 的可能值，逐个计算 $R=R'M^r \bmod N$，根据第(5)步的结果，必然存在一个 r，满足 $d = r + \sum_i d_i$，所以这个 r 也将使得下式成立，即

$$R = R'M^r = M^{(\sum_i d_i)+r} = M^d \bmod N$$

所以，只需要尝试 R 是否能被公钥 e 正确解密，就可以判断这个 r 是否正确。至多尝试 $n+1$ 次后，就可以找出 r 的正确值。

(7) 对于 Share Server 而言，d_i 就是其拥有的部分私钥。对于 Combiner 而言，需要针对不同的 t 个 Share Server 组合，根据 14.4.2 小节的要求计算出对应的 c_j。设 t 个 Share Server 为 $S_1, S_2, \cdots, S_t$，Combiner 为 S_x，下面以这个组合为例说明如何计算出对应的 c_j。

(8) S_x 随机产生一个大整数 E_0，然后将 E_0 发送给 S_{t+1}，S_{t+1} 也产生一个随机大整数 F_1，并计算 $E_1=E_0+F_1+d_{t+1}$，然后 S_{t+1} 再把 E_1 发送给 S_{t+2}，把 F_1 发送给 S_x；S_{t+2} 同样计算 $E_2=E_1+F_2+d_{t+2}$，…，以此类推，S_n 最后将得到

$$E = E_0 + \sum_{i=t+1}^{n} d_i + \sum_{i=1}^{n-t} F_i$$

(9) S_n 把得到的 E_n 再发送给 S_x，因为 $d = r + \sum_i d_i$，所以 S_x 就可以自行计算 c_j 的值

$$c_j = r + E - E_0 - \sum_{i=1}^{n-t} F_i = r + \sum_{i=t+1}^{n} d_i = d - \sum_{i=1}^{t} d_i$$

这样就得到了 14.4.2 小节中说明的部分私钥分配结果。

(10) 反复执行第(8)、(9)步，每次都选取不同的 t 个 Share Server 组合和对应的 Combiner，并产生不同的随机大整数 E_0。组合方案仍然使用 Jing-Feng 方案中说明的组合方法，共需反复执行 C_k^t 次。

至此就完成了 CA 私钥的产生和分割过程。

从这个方案可以看出，虽然其(8)、(9)步也要循环 C_k^t 次，循环次数和方案Ⅰ相同，但大大减少了一次循环的计算量。尽管方案Ⅱ在第(6)步中需要反复重试以寻找 r 的确切值，但整体效率比方案Ⅰ还是有了很大的提高。

14.4.6 签名测试

在 CA 的密钥产生完毕后，整个 CA 系统都进行一次签名测试，签名测试的目的有二：其一是检验产生的 CA 密钥的正确性，这主要是针对 N 的产生过程中 $1/10^{40}$ 的失败概率；其二是使用此次签名结果为以后的部分签名验证做准备。

签名测试的步骤如下。

(1) RA Agent 产生一个随机大整数 w。

(2) RA Agent 在广播信道 B1 上广播 w，这样所有的 Share Sever 都将接收到。

(3) 每台 Share Server 都使用它们的部分私钥计算部分签名 $w_i=w^{d_i} \bmod N$。

(4) 所有的 Share Server 在广播信道 B2 上广播部分签名结果 w_i。

(5) Combiner 接收到所有的 Share Server 部分签名后，针对它所保存的每一个 c_j，都计算一次最终签名 w^d。所有的 Combiner 总共会得到 C_k^t 个结果。

(6) 如果 CA 密钥的产生和分割过程是正确的，则上面的 C_k^t 个结果显然都必须相等，并且可以用公钥 e 验证。因此，此时每台 Combiner 都使用公钥 e 验证它们所得到的计算结果。如果所有的 C_k^t 个结果都验证通过，则此次签名测试通过；否则只要有一个验证不通过，则签名测试失败。

签名测试是 CA 密钥产生和分割的最终决定步骤。若测试通过，则说明 CA 密钥产生和分割过程是正确的，整个过程至此结束；如果测试失败，则整个系统必须恢复到初始状态，重新开始 CA 密钥的产生过程。

14.4.7 密钥产生结果

该方案中，CA 密钥的产生与分割结束后，服务器将获得和保存以下数据。

(1) 每台服务器都要保存其在产生 N 时生成的 p_i 和 q_i。

(2) 每台 Share Server 都保存其部分私钥 d_i。

(3) 对于任意 t 个 Share Server，其对应的 c_j 都由一个 Combiner 保存。

(4) 每个 Combiner 都保存签名测试过程中产生的所有 k 个 w_i。

(5) 所有公开信息，包括 RSA 公钥 N 和 e、签名测试中使用的 w 等。

在以上这些结果中，(1)～(3)步的信息是机密性的，只有保存它们的服务器知道，绝不能向其他服务器或者外界泄露。

14.4.8 部分签名的验证

由于受到攻击部分私钥被篡改，或者由于其他原因某些服务器的部分签名可能会发生错误，CA 系统的入侵容忍性可以保证只要有一定数量的服务器正常工作，整个 CA 系统的运作就仍然可以保持正常。但是查找出发生错误的服务器也是必需的，这有助于系统的排错纠错，及时制止正在发生的入侵事件，提高系统长期运行的安全性和稳定性。

因此还将提供一个方案，使 CA 系统还可以验证服务器的部分签名，以便查找出究竟是哪些服务器发生了错误。文献[12]针对这种情况提出了两种验证协议，将改进其中的一种协议，使其适用于我们提出的入侵容忍 CA 系统。具体协议如下。

执行条件：每个 Combiner 都保存着 14.4.6 小节中产生的 w 和 w_i。假设 CA 系统在计算消息 M 的最终签名时发现错误，R_i 为 Share Server S_i 对 M 的部分签名。

(1) 当一个 Combiner 用公钥验证，发现最终签名有错时，它随机产生两个大整数 u 和 v，并计算

$$M_1 = M^u w^v \bmod N, \quad B_i = R_i^u w_i^v \bmod N$$

(2) Combiner 广播 M_1，要求所有的 Share Server 对其进行部分签名。

(3) 每个 Share Server 都对 M_1 进行部分签名，得到结果 B_i'，并将结果返回给 Combiner。

(4) Combiner 收集到 Share Server 对 M_1 的部分签名 B_i'后，比较 B_i 和 B_i'的值。如果 Share Server 的部分签名是正确的，则显然有

$$B_i' = M_1^{d_i} = M^{d_i u} w^{d_i v} = R_i^u w_i^v = B_i (\bmod N)$$

所以如果 $B_i \neq B_i'$，则对应的 Share Server 必然发生了错误。

(5) 如果 Combiner 没有发现某些 Share Server 的部分签名是错误的，而 RA Agent 又

发现 CA 签名错误，则 RA Agent 可以认为 Combiner 自身发生了错误。

该协议的基本思想借鉴了文献[12]中的一个协议，文献[12]对其协议的可靠性进行了分析，其分析也可用于该协议，总结如下。

(1) 完备性(Completeness)。当 Share Server 并未发生错误，并正确履行该协议时，它也就能通过该协议的检查测试。

(2) 合理性(Soundness)。当一个 Share Server 发生错误时，它没有通过该协议检查的现实可能。实际上，只有一个极其微小的概率使得 Share Server 可以欺骗该协议。

(3) 零知识性(Zeroknoledge)。在协议的整个交互过程中，Share Server 不会泄露其掌握的部分私钥信息。

从以上 3 点分析可以看出，这个部分签名验证协议是足够可靠的，并且不泄露部分私钥信息，仍然满足入侵容忍系统的要求。

14.4.9 部分私钥的更新

当 CA 运行一段时间后，部分私钥泄露的可能性也会自然增加，因此，每隔一段时间全面更新一次所有服务器的部分私钥，可以加强 CA 系统的安全性。这种更新并不改变 CA 的 RSA 密钥对，而只是改变每个服务器所保存的部分私钥值，因此不对 CA 的证书和签名产生任何影响。更新的具体间隔时间取决于具体 CA 的运行策略。

更新过程十分简单，首先为了保证安全性，CA 应停止服务并暂时离线，然后重新运行一遍 14.4.5 小节所描述的 CA 私钥的分割过程，从而获得新的部分私钥分割方案。由于该过程中部分私钥都是随机产生的大整数，跟原有的部分私钥毫无联系，所以这一过程仍然不会泄露 CA 私钥的具体信息。

14.4.10 安全性和性能分析

1. 公钥生成的安全性分析

方案Ⅰ基于简化了的 BGW 多方计算协议，从文献[10]和[11]对 BGW 协议的分析中可以得到，该协议的门限值为 $m=\left\lfloor\frac{n-1}{2}\right\rfloor$。这意味着在方案Ⅰ的生成协议中，即使有 m 个服务器串通共谋，它们也无法获得关于 N 的分解的任何信息，也就无法获得 CA 的根秘密。但是，这样也对整个系统方案的门限值提出了要求，即门限 t 必须满足 $t>m$，在这个条件下，方案Ⅰ是安全的。

关于 N 的合法性检测的安全性前面已经分析过，这里不再赘述。

方案Ⅱ借鉴了文献[13]中的多方计算协议，现有的攻击方法都是针对两方协议的[16,17]，对多方协议($n>2$)还没有造成威胁。

2. 私钥生成的安全性分析

先对方案Ⅰ进行安全性分析。

首先，在协议的前 7 步中，每个服务器向外界透露的唯一数据是 $\gamma_i=\varphi_i+r_ie$，其中 φ_i 在每次计算都是固定的，而 r_i 则是每次随机生成的大整数，所以无论透露了多少个不同的 γ_i，其方程组的变量个数始终比方程数量大 1。即假设有 x 个 $\gamma_i=\varphi_i+r_ie$ 方程，则方程系数矩阵的秩至多为 x，而变量个数为 $x+1$，无法求解。所以外界无法从透露的 γ_i 破解协议。

其次，即使攻击者使用了某种手段，窃取了某个 r_i 的值，也仅仅能从方程 $\gamma_i=\varphi_i+r_ie$ 中获得 φ_i。然而，从 φ_i 也仅仅只能求解出 p_i+q_i，面对两个大整数 p_i 和 q_i 的和，不可能判断出 p_i 和 q_i 的具体值。

最后，分析协议的(8)～(11)步，在这个过程中，即使在最坏的情况下，攻击者最多可以获得 K_i 的值，而为了获得 K_i，显然需要先获得 E_0 的值，所以在最坏的情况下，可以认为 S_j 参与了攻击。然而，即使如此，考虑第(8)步中 K_i 的计算公式：

情况①：$K_i=xr_i-d_i$

由于 r_i 每次随机生成，类似于 γ_i，方程系数矩阵的秩总是比变量数量少，无法求解。

情况②：$K_i=xr_i+y$

情况③：$K_i=xr_i$

以上两种情况下，攻击者可以从 K_i 获得 r_i 的值，由此可知，r_i 的泄露不影响安全性。

此外，K_i 的泄露还能使攻击者获得 c_j 的值，但是由于 S_j 参与了攻击，c_j 的泄露是很自然的事。而且门限系统保证了一个 Combiner 的失控，不会影响整体安全性。

接下来分析方案Ⅱ的安全性。

首先，方案Ⅱ的前 6 步中，对外泄露的主要信息是 $\varphi(N) \bmod e$ 和 $M^{d_i} \bmod N$。后者就是一个正常的部分签名，RSA 算法保证了其安全性。而前者的计算是在模 e 下进行的，考虑到实际使用中 e 的值通常是一个很小的定值，如 65537。在这样小的模数下，要从 $\varphi(N) \bmod e$ 获得确切的 $\varphi(N)$是不可能的。当然，这一点也可以视作对公钥 e 的取值要求。

其次，在方案的(7)～(10)步中，掺杂的随机数 E_0 和 F_i，使得想要从 E_i 获得 d_i，显然也是一件不可能的事情。唯一的可能性需要系统中多个服务器，包括 Combiner 的联合攻击，这一点对系统的门限值有影响。

3. 部分签名验证的安全性分析

对于部分签名验证，只使用了一个额外的数据 $w_i=w^{d_i} \bmod N$。如果攻击者要通过 w_i 获得部分私钥 d_i，实际上就等同于攻击 RSA 算法，因此只要保证随机产生的部分私钥 d_i 足够大，部分签名验证的安全性是可以得到保障的。

4. 部分私钥更新的安全性分析

部分私钥的更新协议，其本质是对私钥产生协议的再运行，从私钥产生协议可以看出，每个服务器的部分私钥是一个随机大整数。因此，在部分私钥更新协议执行前与执行后，服务器所拥有的部分私钥是毫无联系的，被泄露的部分私钥无法对更新后的部分私钥的安全性产生威胁。

5. 效率分析

毫无疑问，相对于原来 Jing-Feng 方案中单独的密钥产生和分发中心，该方案使用的 CA 密钥产生方案的性能会有较大程度的降低。然而，对于一个需要提供稳定服务的 CA 中心来说，其密钥改变的周期是非常长的，除非中途 CA 密钥被泄露出去。因此，CA 密钥的产生协议在很长时间内才会被运行一次，其性能上的降低所带来的影响可以忽略不计。另外，这点影响还换来了 CA 系统更高的安全性，而安全性永远是 CA 系统考虑的第一要素。

在整个方案中，最耗时的无疑是 N 的分布式计算产生。产生的 N 合法性的概率实际

上等于两个随机数 P 和 Q 同时是素数的概率，假设一个随机数是素数的概率为 x，则显然 N 合法的概率只有 x^2。而传统的产生方式都是随机产生 P 和 Q 并检查是否是素数，其随机产生 P 或 Q 是素数的概率都是 x。从这也可以看出，该方案在这一点上效率最低。也就是说，该方案产生 N 所需的循环次数的期望是传统产生方式的平方，而在现有的普通 PC 上，传统 RSA 参数的产生时间一般也不超过 1min，所以该方案的计算时间是可接受的。

另外，对于 CA 运行中的部分私钥更新和部分签名验证，前者也是在较长时间内才运行更新，而后者是要在服务器发生错误时才会运行。因此，对于一个稳定的 CA 来说，这两者都不会影响其正常运行时的性能。可以说，拥有与 Jing-Feng 方案相同的运行性能。

6. 性能的改进考虑

如前所述，在该方案所述的算法和协议中，耗时最多的主要是 CA 密钥 N 的分布式产生。为了尽可能避免无谓的计算，可以在计算 N 之前，先对两个随机数进行筛选，筛除明显非素数的情况，从而加大通过测试协议的概率。考虑到安全性，这样一种筛选也必须分布式进行，且不能泄露 N 的分解信息。

14.5 小结

入侵容忍 CA 协议是构建高安全级别 PKI 系统的核心组件，本章重点介绍了几个典型的入侵容忍 CA 协议。信息安全国家重点实验室 PKI 项目组在这一领域取得了一批创新性工作，包括 Jing-Feng 入侵容忍 CA 协议和自治协同的入侵容忍 CA 协议[15]，自主研制了一套先进的 PKI 系统并得到实际应用，形成了系列国家标准，部分工作已在文献[18]～[20]中做了较为详细的总结。本章主要介绍了我们自己的一些工作，对其他一些有代表性的工作也做了一些综述和对比介绍，感兴趣的读者可从文中介绍的线索参阅相关文献。关于传统 PKI 方面的介绍可参阅文献[21]和[22]。荆继武教授和庄湧博士也参加了本章的写作，作者在此表示衷心的感谢。

参考文献

[1] Wu T, Malkin M, Boneh D. Building intrusion-tolerant applications, Proceedings of USENIX Security Symposium, August 1999, 79～91.

[2] Frankel Y. A practical protocol for large group oriented networks. Eurocrypt 89. 1989, 56～61.

[3] Alon N, Galil Z, Yung M. Dynamic re-sharing verifiable secret sharing against a mobile adversary, in Proceedings of the 1995 European Symposium on Algorithms (ESA). 1995, 523～537.

[4] Desmedt Y, Crescenzo G D, Burmester M. Multiplicative non-abelian sharing schemes and their application to threshold cryptography, Proceedings ASIACRYPT'94. 1994, 21～32.

[5] Frankel Y, Gemmel P, MacKenzie P, Yung M. Optimal-resilience proactive public-key cryptosystems. Proceedings FOCS'97. 1997, 384～393.

[6] Shoup V, Practical threshold signatures, in Proc. Eurocrypt 2000, Bruges (Brugge), Belgium: Springer, 2000, 207～220.

[7] 荆继武，冯登国. 一种入侵容忍的 CA 方案. 软件学报，2002. 13(8): 1417～1422.

[8] Jing J W, Liu P, Feng D G. ARECA: A highly attack resilient Certification Authority. Proceedings of the ACM Workshop on Survivable and Self-Regenerative Systems (In Association with 10th ACM

Conference on Computer Communications Security), 2003, 53～63.

[9] Feng D G, Xiang J. Experience on intrusion tolerance distributed systems, Proceedings of the 29th Annual International Computer Software and Applications Conference. 270～271, 2005.

[10] Boneh D, Franklin M. Efficient Generation of Shared RSA Keys, in Advances in Cryptology—Crypto'97. LNCS 1294. Springer-Verlag Heidelberg. 1997, 425～439.

[11] Malkin M, Wu T, Boneh D. Experimenting with Shared Generation of RSA keys. Internet Society's 1999 Symposium on Network and Distributed System Security (SNDSS). 1999, 43～56.

[12] Gennaro R. Robust and Efficient Sharing of RSA Functions[A]. Advances in Cryptology-CRYPTO'96. LNCS 1109[C]. Springer-Verlag Heidelberg, 1996, 157～172.

[13] Cocks C. Split Generation of RSA Parameters with Multiple Participants. Technical report. Available at http://www.cesg.gov.uk. 1998.

[14] Catalano D, Gennaro R, Halevi S. Computing inverses over a shared secret modulus. Eurocrypt'00, LNCS 1807. Springer-Verlag Heidelberg, 2000, 190～206.

[15] 庄湧,冯登国.一种针对弹性 CA 的分布式密钥产生方案,计算机研究与发展,2007,44(02),230～235.

[16] Blackburn S. Shared Generation of Shared RSA Keys. Technical Report CORR 98～119. 1998.

[17] Joye M, Pinch R. Cheating in split-knowledge RSA parameter generation. in Proc. Workshop on Coding and Cryptography, Paris, France. 1999. 157～163.

[18] 荆继武.高安全 PKI 系统研究,中国科学院研究生院博士学位论文,2003.

[19] 庄湧.高安全等级 PKI 系统关键技术研究,中国科学院软件研究所博士学位论文,2006.

[20] Zhang L W, Feng D G. Intrusion Tolerant CA Scheme with Cheaters Detection Ability. ISPA Workshops 2005: 378～386.

[21] Adams C, Lloyd S. Understanding Publickey Infrastructure: Concepts, Standars, and Deployment Consideratiods, Macmillan Technical Publishing, 1999(公开密钥基础设施——概念、标准和实施.冯登国等译.北京:人民邮电出版社,2001.)

[22] 荆继武,林璟锵,冯登国.PKI 技术.北京:科学出版社,2008.

第 15 章　基于身份的 PKI 协议

基于身份的公钥密码体制(简称 ID-PKC,又称基于标识的公钥密码体制)近年来得到了迅速发展,并成为一个研究热点,其基本思想是以任意字符串如姓名、E-mail 地址等作为用户的公钥,由一个可信的私钥生成中心(Public Key Generator,PKG)为其生成私钥。与传统的公钥密码体制相比,ID-PKC 最突出的特点在于公钥具有语义性。传统公钥密码体制中的公钥通常是随机产生的二进制串,其本身不具有任何语义;而在 ID-PKC 中,公钥可以是事先选定的任何有意义的字符串,如身份信息。由于可以使用公钥直接表达身份信息,ID-PKC 不再需要用证书绑定用户身份和公钥。

ID-PKC 的思想最早是由 Shamir 提出的[1],其目的是避免传统公钥密码体制中由证书引起的存储、查询等问题。Shamir 给出了第一个基于身份的签名方案(Identity-Based Signature,IBS)[1]。此后出现了多种基于身份的签名方案和身份识别方案(Identity-Based Identification,IBI)[2,3]。然而,从 Shamir 提出 ID-PKC 思想后的很长一段时间内,一直没有一个有效且安全的基于身份的加密方案(Identity-Based Encryption,IBE)出现,这在很大程度上影响了 ID-PKC 的研究和应用。1998 年 Cocks提出了一个基于二次剩余的 IBE 方案[4],但该方案逐比特加、解密,效率太低,不能满足实际需要。直到 2001 年,才由 Boneh 和 Franklin 提出了第一个能够满足实际应用的 IBE 方案[5]。Boneh-Franklin IBE 方案的出现极大地推动了 ID-PKC 的研究和发展,涌现出一大批 ID-PKC 方案。

ID-PKC 的发展使得在此基础上构建基于身份的 PKI 系统(简称 ID-PKI)成为可能。得益于 ID-PKC 的特性,ID-PKI 在许多方面比传统 PKI 更具有优势。首先,由于不必再为用户颁发证书,ID-PKI 系统可以省去证书库、LDAP 等系统部件的建设和维护成本,因而更加轻量、灵活。其次,由于公钥可从身份信息直接计算得出,通信各方在相互认证时不必再事先传递、查询证书,因而可以较大地降低通信量并减少交互次数。再其次,由于用户的公钥即身份是天然存在的,即使用户还没有向密钥中心申请私钥,依赖方(Relying Party)也可以先用其公钥加密消息,接收方可以在收到加密消息后再去申请私钥。而在传统 PKI 中,依赖方必须等接收方申请证书后,才能使用其证书中的公钥加密消息。

ID-PKI 的上述优势使其具有良好的应用前景,被视为未来 PKI 的一个发展方向[6]。然而应该注意到,ID-PKI 在具有显著优势的同时,也还存在若干问题,这些问题直接影响了 ID-PKI 的推广和应用。其中最主要的是 ID-PKI 中的密钥管理问题,包括私钥的安全分发、密钥撤销及密钥托管等。首先,由于 ID-PKI 中所有用户的私钥只能由密钥中心产生,因此必须以一种安全的方式把私钥从密钥中心分发到用户手中,保证途中用户私钥的机密性和完整性。其次,同传统 PKI 类似,ID-PKI 中同样需要密钥撤销。但由于 ID-PKI 独特的性质,传统 PKI 中的证书撤销方法如 CRL[7]、OCSP[8] 等在 ID-PKI 中并不适用。再其次,由于密钥中心掌握所有用户的完整私钥,密钥托管成为 ID-PKI 与生俱来的问题。虽然在某些情形下,密钥托管是有益的,但在多数应用中,密钥托管是需要避免的。如何避免 ID-PKI 中的密钥托管也成为需要解决的一个问题。虽然针对 ID-PKI 密钥管理问题目前

已经有一些研究成果，但还没有能够同时解决上述 3 个问题的解决方案，而且不同方案也很难一起使用，因此在 ID-PKI 密钥管理问题上值得进一步研究。除了上述密钥管理问题外，ID-PKI 的另一个重要问题是如何在具体环境中应用 ID-PKI，充分发挥 ID-PKI 的优势。然而，目前的研究大多集中在基础 ID-PKC 方案上，对这类问题的研究仍比较少。我们针对这些问题进行了系统深入研究，并取得一批新成果，本章主要介绍了我们在这一领域的一些研究工作。

15.1　ID-PKC 方案简介

ID-PKC 起源于 Shamir 在文献[1]中提出的第一个基于身份的签名方案。之后 ID-PKC 经历了较长时间的缓慢发展。2000 年 Sakai、Ohgishi 和 Kasahara 提出了基于双线性映射的 IBS 方案(简称为 SOK-IBS 方案)[9]。2001 年 Boneh 和 Franklin 提出了基于双线性映射的 IBE 方案[5]，该方案的提出是 ID-PKC 发展过程中的一个重要里程碑。此后，ID-PKC 得到了迅速发展，出现了大量的相关研究成果，包括两个 SOK-IBS 改进方案[10,11]、基于身份的加密方案[12~14]、基于身份的签名方案[15,16]、基于身份的层次加密方案[17]、基于身份的签密方案[18~21]及基于身份的密钥协商协议[22~25]等。相关方案可以参见综述文献[26]、[27]。

最典型的 ID-PKC 方案有 Shamir 提出的 IBS 方案、Boneh 和 Franklin 提出的 IBE 方案，以及 Sakai、Ohgishi 和 Kasahara 提出的 IBS 方案。Shamir IBS 方案在 9.8 节已做了详细介绍，Shamir 在其论文中没有给出该方案的安全性证明，Libert 等人在文献[11]中证明了该方案在选择消息攻击下是存在性非伪造的。本节重点介绍后两个方案。

15.1.1　几个重要概念

定义 15.1(双线性映射，Bilinear map)　设 G_1 为阶为 q 的加法循环群，G_2 为阶为 q 的乘法循环群，q 是一个大素数。定义映射 $\hat{e}: G_1 \times G_1 \to G_2$，如果 $\hat{e}$ 满足以下性质，则称 $\hat{e}$ 为一个可接受的双线性映射。

(1) 双线性性(Bilinear)。对所有的 $P,Q \in G_1$，$a,b \in Z$，都有 $\hat{e}(aP,bQ)=\hat{e}(P,Q)^{ab}$。

(2) 非退化性(Non-degenerate)。存在 $P,Q \in G_1$，使得 $\hat{e}(P,Q) \neq 1_{G_2}$，其中 1_{G_2} 为群 G_2 的单位元。

(3) 可计算性(Computable)。存在一个算法，对任何 $P,Q \in G_1$ 都可以有效地计算出 $\hat{e}(P,Q)$。

双线性映射可以从椭圆曲线上的 Weil 对(Weil Pairing)和 Tate 对(Tate Pairing)中得到。更多内容可参阅文献[5]和[28]。

CDH 问题和 CDH 假设虽然已在前面作过介绍，但这里再回顾一下椭圆曲线群上的相关定义。

定义 15.2(CDH 问题)　令 G 为 q 阶的循环群，q 是一个大素数。P 为 G 的任一生成元。CDH 问题是：对于任意未知正整数 $a,b \in Z_q$，给定 aP 和 bP，计算 abP。

定义 15.3(CDH 假设)　令 PGen 为 CDH 参数生成器，即输入安全参数 1^k，在多项式时间内输出一个素数 q 阶的循环群 G，以及 G 上的一个生成元 $P \in G^*$。定义算法 A 在解决

PGen 生成的 CDH 问题上的优势为

$$\mathrm{Adv}^{\mathrm{CDH}}_{\mathrm{PGen},A}(1^k) = \Pr[A(G,P,aP,bP) = abP \mid \langle G,P\rangle \leftarrow \mathrm{PGen}(1^k), a,b \xleftarrow{R} Z_q^*]$$

若对任意概率多项式时间算法 A(以 k 为参数),$\mathrm{Adv}^{\mathrm{CDH}}_{\mathrm{PGen},A}(k)$都是可忽略的,则称 PGen 满足 CDH 假设。

定义 15.4(BDH 问题) 令 q 为一个大素数,G_1 为 q 阶的加法循环群,G_2 为 q 阶的乘法循环群,P 为 G_1 的生成元,映射 $\hat{e}:G_1\times G_1\rightarrow G_2$ 为可接受的双线性映射,定义在$\langle G_1,G_2,\hat{e}\rangle$上的 BDH 问题为:给定$\langle P,aP,bP,cP\rangle$,其中 $a,b,c\in Z_q^*$,计算 $W=\hat{e}(P,P)^{abc}\in G_2$。

定义 15.5(BDH 假设) 令 PGen 为 BDH 问题参数生成器,即输入安全参数 1^k,在多项式时间内输出阶均为素数 q 的加法循环群 G_1 和乘法循环群 G_2,双线性映射 $\hat{e}:G_1\times G_1\rightarrow G_2$,以及 G_1 的生成元 P。对任意的多项式时间算法 A,定义其在解决 PGen 生成的 BDH 问题上的优势为

$$\mathrm{Adv}^{\mathrm{BDH}}_{\mathrm{PGen},A}(1^k) = \Pr[A(q,G_1,G_2,\hat{e},P,aP,bP,cP) = \hat{e}(P,P)^{abc} \mid \langle G_1,G_2,\hat{e},P\rangle \leftarrow \mathrm{PGen}(1^k), a,b,c \leftarrow_R Z_q^*]$$

若对于任意概率多项式时间算法 A(以 k 为参数),$\mathrm{Adv}^{\mathrm{BDH}}_{\mathrm{PGen},A}(k)$都是可忽略的,则称 PGen 满足 BDH 假设。

定义 15.6(q-SDH 假设) 令(G_1,G_2)都是阶为素数 q 的循环群,对所有的概率多项式时间算法 A,概率 $\Pr[A((g_1,g_2,g_2^{\gamma},g_2^{\gamma^2},\cdots,g_2^{\gamma^t})\in(G_1\times(G_2)^{t+1}))=(g_1^{1/(\gamma+x)},x)\wedge x\in Z_q^*]$ 是可忽略的。其中 $x,\gamma\in Z_q^*$,ψ 是从群 G_2 到 G_1 的同构,且 $\psi(g_2)=g_1$。

15.1.2 Boneh-Franklin IBE 方案

文献[5]中给出了两个 IBE 方案,即基本方案(BasicIdent)和完整方案(FullIdent)。其中 BasicIdent 方案在 RO 模型中被证明是选择明文攻击语义安全的(IND-ID-CPA),FullIdent 方案是在 BasicIdent 基础上使用了 Fujisaki-Okamoto 的变换方法[29]得到的选择密文攻击语义安全(IND-ID-CCA)的改进方案。下面分别介绍这两个方案。

1. BasicIdent 方案

BasicIdent 方案包含 4 个过程:初始化、生成密钥、加密和解密,具体过程如下。

(1) 初始化。根据输入的安全参数 k,生成一个素数 q,q 阶的加法群 G_1 和 q 阶的乘法群 G_2,双线性映射 $\hat{e}:G_1\times G_1\rightarrow G_2$ 以及 G_1 的生成元 P。选择两个 Hash 函数:$H_1:\{0,1\}^*\rightarrow G_1^*$ 和 $H_2:G_2^*\rightarrow\{0,1\}^n$。选择一个随机数 $s\in_R Z_q^*$,计算 $P_{\mathrm{Pub}}=sP$。系统的公开参数为$\{q, G_1, G_2, P, H_1, H_2, P_{\mathrm{Pub}}\}$,系统的主密钥为 s,由 PKG 独自保管。

(2) 生成密钥。对一个输入的字符串 $\mathrm{ID}\in\{0,1\}^*$,其公钥为 $Q_{\mathrm{ID}}=H_1(\mathrm{ID})\in G_1^*$,其对应的私钥为 $d_{\mathrm{ID}}=sQ_{\mathrm{ID}}$。

(3) 加密。使用字符串 ID 加密消息 $m\in\{0,1\}^n$ 的步骤为:首先计算 $Q_{\mathrm{ID}}=H_1(\mathrm{ID})\in G_1^*$,其次选择随机数 $r\in Z_q^*$,最后计算 $U=\mathrm{r}\mathrm{P}$,$V=m\oplus H_2(g_{\mathrm{ID}}^r)$,其中 $g_{\mathrm{ID}}=\hat{e}(Q_{\mathrm{ID}},P_{\mathrm{Pub}})$。密文为 $c=<U,V>$。

(4) 解密。收到密文 $c=<U,V>$后,接收者使用自己的私钥 $d_{\mathrm{ID}}=sQ_{\mathrm{ID}}$计算明文消息

$m=V\oplus H_2(d_{ID},U)$。

2. FullIdent 方案

Boneh 和 Franklin 在 BasicIdent 方案基础上使用 Fujisaki 和 Okamoto 提出的转换方法[29]得到 FullIdent 方案。Fujisaki-Okamoto 方法通过在密文中增加密文合法性检查信息，把具有单向安全性的加密方案(One-way encryption)转换为具有选择密文攻击下语义安全性(IND-CCA)的加密方案。FullIdnet 方案与 BasicIdent 基本相同，具体过程如下。

(1) 初始化。与 BasicIdent 基本相同，但增加了两个 Hash 函数 $H_3:\{0,1\}^n\times\{0,1\}^n\rightarrow Z_q^*$ 和 $H_4:\{0,1\}^n\rightarrow\{0,1\}^n$。

(2) 生成密钥。与 BasicIdent 完全相同。

(3) 加密。使用字符串 ID 加密消息 $m\in\{0,1\}^n$ 的步骤为：首先计算 $Q_{ID}=H_1(ID)\in G_1^*$，其次选择随机数 $\sigma\in\{0,1\}^n$，最后令 $r=H_3(\sigma,m)$，密文为

$$c=<rP,\sigma\oplus H_2(g_{ID}^r),m\oplus H_4(\sigma)>,\quad \text{其中 } g_{ID}=\hat{e}(Q_{ID},P_{Pub})$$

(4) 解密。收到密文 $c=<U,V,W>$ 后，解密步骤为：首先使用私钥 d_{ID} 计算 $\sigma=V\oplus H_2(\hat{e}(d_{ID},U))$，其次计算 $m=W\oplus H_4(\sigma)$，最后令 $r=H_3(\sigma,m)$，比较 U 和 rP 是否相等，如果是则接受明文 m，否则丢弃该密文。

Boneh 和 Franklin 在 RO 模型中证明了这两个方案的安全性。需要指出的是，Boneh 和 Franklin 对攻击者能力的形式化定义(IND-ID-CPA 和 IND-ID-CCA)与标准定义(IND-CPA 和 IND-CCA)[30,31]是有区别的，即在标准定义基础上增强了攻击者能力，使其可以询问用户私钥。这是为了更好地模拟实际环境下攻击者的能力。在 ID-PKC 中，同一个 PKG 下所有用户的私钥都由同一个主密钥生成的，实际环境中攻击者有可能已经掌握了某些用户的私钥，利用这些已知信息再去攻击同一个 PKG 的其他用户，因此必须防止攻击者利用用户私钥间的相关性构造对 IBE 方案的攻击。Boneh-Franklin IBE 方案之后的 ID-PKC 密码方案在进行安全性证明时也都考虑了这一点。

15.1.3 SOK-IBS 原始方案

本节介绍 SOK-IBS 的原始方案[9]和两个改进方案[10,11]。SOK-IBS 原始方案的具体过程如下。

(1) 初始化。对输入的安全参数 1^k，PKG 选择两个阶为素数 q 的群 G_1 和 $G_2(q>2^k)$，一个双线性映射 $\hat{e}:G_1\times G_1\rightarrow G_2$，$G_1$ 的生成元 P，一个随机选择的主密钥 $s\in Z_q^*$ 和一个相应的公钥 $P_{Pub}=sP\in G_1$。PKG 还选择两个 Hash 函数：$H_1,H_2:\{0,1\}^n\rightarrow G_1^*$。系统的公开参数为：params$=\{G_1,G_2,\hat{e},P,P_{Pub},H_1,H_2\}$。

(2) 生成私钥。对用户身份信息 ID，PKG 计算 $Q_{ID}=H_1(ID)\in G_1$，则用户的私钥为 $d_{ID}=sQ_{ID}\in G_1$。

(3) 签名。用户 ID 对消息 M 的签名过程如下。

① 选择随机数 $r\in_R Z_q^*$，计算 $U=rP\in G_1$ 和 $H=H_2(M)\in G_1$。

② 计算 $V=d_{ID}+rH\in G_1$。

签名结果为 $\sigma=<U,V>\in G_1\times G_1$。

(4) 验证。为了验证用户 ID 对消息 M 的签名结果 $\sigma=<U,V>\in G_1\times G_1$，验证者首先计

算 $Q_{ID}=H_1(ID)\in G_1$ 和 $H=H_2(M)\in G_1$。然后判断等式 $\hat{e}(V,P)=\hat{e}(Q_{ID},P_{Pub})\cdot\hat{e}(H,U)$是否成立。如果成立，则接受该签名结果；否则，拒绝接受。

SOK-IBS 原始方案同两个改进方案[10,11]的区别仅在于 H 的赋值。SOK-IBS-1 方案[10]中 H 被设置为 $H_2(M,U)$，SOK-IBS-2 方案[11]中 H 被设置为 $H_2(ID,M,U)$。显然，对 H 的修改并不影响方案的效率。

SOK-IBS 原始方案是一个很有意思的方案。与大多数其他 IBS 方案[1,2,24,25]以及两个 SOK-IBS 改进方案[10,11]不同的是，SOK-IBS 原始方案不能看做是使用 Fiat-Shamir 转换方法[2]对某个识别方案作用的结果。正是由于这一点，文献[10]中的通用框架和文献[32]中的 Forking 引理都很难被用来证明该方案的安全性。文献[33]中证明了该方案在选择消息攻击下是能够抵抗存在性伪造的(UF-CMA)，并说明了 SOK-IBS 原始方案不是选择消息攻击下强存在性非伪造安全的(sUF-CMA)，虽然两个改进方案[11,12]具有这样的安全性。从这一点上可以看出，文献[10]对 SOK-IBS 方案的改进是很有意义的，在不影响运算效率的同时提高了安全性。当然，在大多数的应用环境中，UF-CMA 安全性已足够。

15.2 基于身份的密钥隔离密码方案

密钥泄露可能是对密码系统最具灾难性的破坏，因为一旦密钥被攻击者窃取，即使密码算法以及所基于的安全假设都是安全的，整个密码系统也不再安全。在实际环境中，攻击者通常也会选择窃取用户的私钥，而不是直接攻击密码算法或安全假设，因为前者相对而言更容易。正因为如此，人们一直在研究如何避免密钥泄露造成的危害。简单来讲，目前的研究思路可以大致分为两种[34]。一种思路是防止密钥被窃取，这类方法通常使用防窜扰密码模块，如 IC 卡等存放密钥。这种方法可以在很大程度上防止攻击者获取密钥，但又面临着密钥设备(特别是便携设备)被窃取的问题。另外，这种方法通常对硬件设备有较高的要求，对普通应用来说成本过高。另一种思路是接受密钥容易被窃取这一现实，转而考虑如何降低密钥被窃取带来的危害。目前关于这类方法的研究成果比较多，如秘密共享方法[35,36]、门限密码方法[37~39]、前摄密码方法[40~43]、前向安全密码方法[44,45]、密钥隔离密码方法[34,46,47]和入侵容忍密码方法[48,49]等。

密钥隔离密码方案(Key-Insulated Cryptography)最早是由 Dodis 等人提出的[34]，其基本思想是把用户私钥拆分成两部分，一部分存放在安全性较差的密码运算设备中(如用户的笔记本电脑)，另一部分存放在一个被称为 Base 的物理安全的设备中(如家中的台式机)。本节主要论述如何把密钥隔离的思想引入到 ID-PKC 之中。首先给出了基本的基于身份的密钥隔离密码方案(Identity-Based Key-Insulated Cryptography，IDKIC)，包含一个加密方案(IDKIE)和一个签名方案(IDKIS)。然后针对密钥隔离密码方案中的信道安全问题提出了解决方案，提出了一种可抵抗主动攻击者的 IDKIE 方案(Active-Adversary-Resistant IDKIE，AR-IDKIE)。在 RO 模型下，可证明上述方案的安全性。

15.2.1 基于身份的密钥隔离加密方案

本节主要介绍一种基于身份的密钥隔离加密方案(Identity-Based Key-Insulated Encryption Scheme，IDKIE)。

1. IDKIE 模型

同其他密钥隔离密码方案类似，IDKIE 中用户的私钥被拆分成两部分，由两个设备分别持有：一个设备是用户的密码运算设备（如用户的笔记本电脑），为方便起见称之为 User；另一个设备存放用户的另一部分私钥，定期产生密钥更新信息（如用户存放在家中的台式机），称之为 Server。称 User 和 Server 持有的用户部分私钥为 Home key，分别记为 $Sk_{ID,U}$ 和 $Sk_{ID,S}$。IDKIE 方案的整个生命周期被划分为多个连续的时间段，如 2009 年12 月、2010 年 1 月、2010 年 2 月，……，记这些时间段为 t_i。在 t_{i-1} 时间段结束时，User 向 Server 索取 t_i 时间段的密钥更新信息，记为 Ku_{ID,t_i}。使用 Home key $Sk_{ID,U}$ 和密钥更新信息 Ku_{ID,t_i}，User 可以计算出在 t_i 时间段的临时私钥，记为 Tk_{ID,t_i}，用户生成新的临时私钥后立即删除之前的临时私钥。User 使用 Tk_{ID,t_i} 可以在 t_i 时间段完全自主地解密密文。

方案中各实体之间的关系如图 15.1 所示。

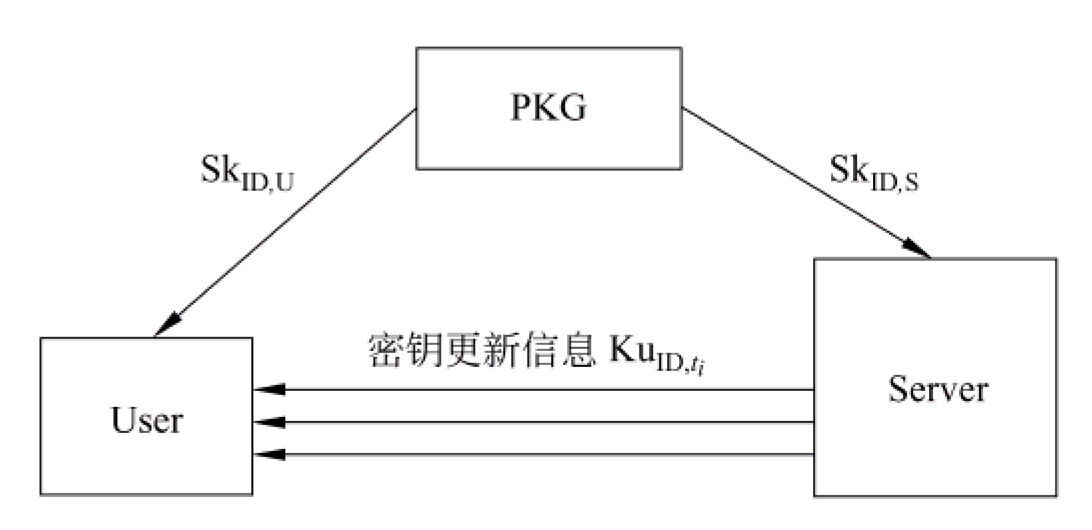


图 15.1　方案中各实体之间的模型

同传统密钥隔离密码方案类似，在 IDKIE 方案中攻击者可以多次侵入 User 设备，并获得其中的 Home key 和临时私钥。但攻击者只能获得其侵入时的用户临时私钥，其他时间段的临时私钥仍是安全的（总假设用户会立即删除旧时间段的临时私钥）。攻击者还可以获得 Server 端的 Home key，但仅获得 Server 端的 Home key 或仅截获密钥更新信息不会影响任何时间段用户私钥的安全性。ID-PKC 的特殊性质还使得攻击者可以与恶意用户串通，但掌握再多用户的私钥，也无助于攻击者攻击其他合法用户的私钥。当然，如果在用户私钥生命周期内，攻击者既攻破了 User 端也攻破了 Server 端，那么就可以获得用户的完整私钥；或者如果获得 User 端的 Home key $Sk_{ID,U}$ 后又可以截获以后的密钥更新信息，攻击者就可以获得以后时间段的用户临时私钥。正是由于这个原因，通常密钥隔离密码方案都假设 Server 是足够安全的（存放在家中），或者 User 和 Server 的交互是通过安全信道进行的。对于通过非安全信道进行密钥更新的情况将在 15.2.3 小节讨论，并给出相应的解决方案。

定义 15.7　IDKIE 是一个由以下 6 个函数组成的六元组 IDKIE：=｛PGen，KGen，UpdGen，UserUpd，Enc，Dec｝。

（1）PGen：公开参数生成函数。输入安全参数 1^k，输出系统公开参数 params 和主密钥（Master key）s。

（2）KGen：用户私钥初始生成函数。输入用户身份 ID 和系统公开参数 params，输出用户的 Home key｛$Sk_{ID,U}$，$Sk_{ID,S}$｝。

（3）UpdGen：密钥更新信息生成函数。输入 Server 端 Home key $Sk_{ID,S}$，时间段 t 和公开参数 params，输出 t 时间段的密钥更新信息 $Ku_{ID,t}$。

（4）UserUpd：用户密钥更新函数。输入 User 端 Home key $Sk_{ID,U}$，时间段 t，密钥更新信息 $Ku_{ID,t}$ 和公开参数 params，输出用户在 t 时间段的临时私钥 $Tk_{ID,t}$ 或者输出错误标记 ⊥。如果成功生成 $Tk_{ID,t}$，则立即删除上个时间段的临时私钥。

（5）Enc：加密函数。输入用户身份信息 ID、时间段 t、明文 M 和公开参数 params，输出密文 $<t,C>$。

（6）Dec：解密函数。输入密文 $<t,C>$，t 时间段的临时私钥 $Tk_{ID,t}$ 和公开参数 params，输出明文 M 或者输出错误标记 ⊥。

PGen 函数由 PKG 在系统初始化时运行，在整个系统生命周期内仅运行一次。KGen 函数由 PKG 在每个用户初始注册时运行，生成用户的 Home key，之后用户自行维护密钥，不需要 PKG 参与。UpdGen 函数是在每个时间段结束时（根据用户需要也可以在任意时刻）由 Server 运行的，用于生成 User 的密钥更新信息。Enc 由依赖方使用，用于产生密文，密文中包含了加密时的时间段信息。UserUpd 函数和 Dec 函数均由 User 运行，分别用于更新临时私钥和解密密文。

IDKIE 模型中，假设攻击者只能多项式次（Polynomial Times）地攻破 User 端并获得 Home key 和临时私钥。在这种假设下研究 IDKIE 方案的安全性。令 A 是 IDKIE 方案的一个概率多项式时间攻击者，为了模仿实际环境中攻击者的行为，允许 A 在与挑战者的游戏中询问以下 Oracle。

（1）Home key Oracle O_{Hk}：输入用户的身份信息 ID 和 $P\in\{U,S\}$（U 表示 User，S 表示 Server），输出对应的 Home key $Sk_{ID,P}$。

（2）Temp key Oracle O_{Tk}：输入用户身份信息 ID 和时间段 t，输出对应的临时私钥 $Tk_{ID,t}$。

（3）Dec Oracle O_{Dec}：输入用户身份信息 ID 和密文 $<t,C>$，输出使用临时私钥 $Tk_{ID,t}$ 对 C 的解密结果。

（4）L-R Oracle O_{LR}：输入用户身份信息 ID^*，时间段 t^* 和两个等长的明文 m_0、m_1，输出挑战密文 $C^*=Enc(ID^*,t^*,m_b)$，其中 $b\in\{0,1\}$ 是事先选定的。攻击者猜测 b 的值。

此处没有直接为攻击者提供密钥更新信息 Oracle，而是使用 Home key Oracle 和 Temp key Oracle 模拟这种攻击能力。如果攻击者已经掌握了 Server 端 Home key，那么它可以自行生成任何时间段的密钥更新信息，这种情况下获取密钥更新信息的行为可以用 $O_{Hk}(ID,S)$ 模拟。如果攻击者已经掌握了 User 端的 Home key，其获取密钥更新的行为可以通过询问 $O_{Hk}(ID,U)$ 和 $O_{Tk}(ID,t)$ 模拟。如果攻击者不掌握任何一方的 Home key，显然其攻击能力完全包含在前两种攻击者能力中，因此，可以将其强化为上述任一种攻击者。

为了表述方便，可以把 O_{Hk}、O_{Tk} 统一表示成 O_{Sec}，并使用不同的字符串输入区分，如 $O_{Sec}(Hk,ID,P)$ 表示 $O_{Hk}(ID,P)$。

对一个 O_{Sec} 询问集合 Q，称 Home key $Sk_{ID,P}$ 是 Q-exposed 的，如果 $O_{Sec}(Hk,ID,P)\in Q$；称临时私钥 $Tk_{ID,t}$ 是 Q-exposed 的，如果 $O_{Sec}(Tk,ID,t)\in Q$。如果临时私钥 $Tk_{ID,t}$ 是 Q-exposed 的，或者 Home key $Sk_{ID,U}$ 和 $Sk_{ID,S}$ 都是 Q-exposed 的，则称该 IDKIE 方案是 (ID,t,Q)-compromised 的。

在游戏中，允许攻击者随意访问多项式次数的上述各种 Oracle（但仅访问 L-R Oracle 一次），如果攻击者最终在 L-R Oracle 询问 $O_{LR}(ID^*,t^*,\cdot)$ 中输出了对 b 正确的猜测 b'，

使得 $b=b'$，并且满足以下条件：该 IDKIE 方案不是(ID^*，t^*，Q)-compromised 的；设 O_{LR} 返回的密文为 C^*，攻击者没向 O_{Dec} 询问过 $O_{Dec}(ID^*,t^*,C^*)$，则称攻击者赢得了与挑战者的游戏。攻击者获胜的优势定义为

$$\mathrm{Adv}_A(k)=|\Pr[b=b']-1/2|$$

定义 15.8 称 IDKIE 方案在自适应选择密文攻击下是语义安全的(IDKIE-CCA)，如果对任意概率多项式时间攻击者 A，在上述游戏中的优势 $\mathrm{Adv}_A(k)$ 都是可忽略的。

2. 基于双线性映射的 IDKIE 方案

这里基于椭圆曲线上的双线性映射，构造了一个新的 IDKIE 方案。该方案可以看做是以 Boneh-Franklin IBE 方案[5]为基础修改而来的，在继承 Boneh-Franklin IBE 的参数空间的基础上，增加了 Hash 函数 H_u。方案生成的密文消息比 Boneh-Franklin IBE 方案多了 $T=rHu$ 和时间段 t，其他密文内容与其完全一致。IDKIE 方案的具体过程如下。

(1) PGen。输入安全参数 1^k，PKG 生成 BDHP 参数($q,G_1,G_2,\hat{e}$)，选择 G_1 上的任意生成元 $P\in G_1$，随机选择 $s\in_R Z_q^*$，计算 $P_{Pub}=sP$。选择 Hash 函数 $H_1:\{0,1\}^*\to G_1$，$H_2:G_2\to\{0,1\}^n$，$H_3:\{0,1\}^n\times\{0,1\}^n\to Z_q^*$，$H_4:\{0,1\}^n\to\{0,1\}^n$ 和 $H_u:\{0,1\}^*\to G_1$，其中 n 是明文长度。系统公开参数 params$=\{G_1,G_2,\hat{e},P,P_{Pub},H_1,H_2,H_3,H_4,H_u\}$，系统主密钥为 s。

(2) KGen。输入主密钥 s，系统公开参数 params 和用户身份信息 ID，计算 $Q_{ID}=H_1(ID)\in G_1$，$d_{ID}=sQ_{ID}\in G_1$。随机选择 $Sk_{ID,U}\in_R G_1$，令 $Sk_{ID,S}=d_{ID}-Sk_{ID,U}$，输出 $\{Sk_{ID,U},Sk_{ID,S}\}$。

(3) UpdGen。输入 Server 的 Home key $Sk_{ID,S}$，时间段 t 和系统公开参数 params，Server 计算 $Hu=H_u(t)$，随机选择 $x\in_R Z_q^*$，计算 $X=Sk_{ID,S}+xHu\in G_1$ 和 $Y=xP\in G_1$，返回 $Ku_{ID,t}=\langle X,Y\rangle$。

(4) UserUpd。输入 User 的 Home key $Sk_{ID,U}$、时间段 t、密钥更新信息 $Ku_{ID,t}=\langle X,Y\rangle$ 和系统公开参数 params，User 首先检查等式 $\hat{e}(Sk_{ID,U}+X,P)=\hat{e}(Q_{ID},P_{Pub})\cdot\hat{e}(Hu(t),Y)$ 是否成立。如果不成立，返回错误标记⊥；否则，令临时私钥 $Tk_{ID,t}=\langle Sk_{ID,U}+X,Y\rangle$。

(5) Enc。输入用户身份信息 ID、时间段 t、明文 M 和系统公开参数 params，计算 $Q_{ID}=H_1(ID)$，随机选择 $\sigma\in_R\{0,1\}^n$，计算 $r=H_3(\sigma,M)$，$Hu=H_u(t)$ 和 $g=\hat{e}(Q_{ID},P_{Pub})$。$C=\langle rHu,rP,\sigma\oplus H_2(g^r),M\oplus H_4(\sigma)\rangle$。返回 $\langle t,C\rangle$。

(6) Dec。输入密文 $\langle t,C\rangle$ 其中 $C=\langle T,U,V,W\rangle$，临时私钥 $Tk_{ID,t}=\langle X_{ID,t},Y_{ID,t}\rangle$ 和系统公开参数 params，首先计算 $\sigma=V\oplus H_2\left(\dfrac{\hat{e}(X_{ID,t},U)}{\hat{e}(T,Y_{ID,t})}\right)$，$M=H_4(\sigma)$，其次令 $r=H_3(\sigma,M)$，检查是否 $U=rP$。如果不等，返回错误标记⊥；否则，返回 M。

IDKIE 方案的正确性很容易验证，因为有等式

$$\frac{\hat{e}(X_{ID,t},U)}{\hat{e}(T,Y_{ID,t})}=\frac{\hat{e}(d_{ID}+xHu,rP)}{\hat{e}(rHu,xP)}=\frac{\hat{e}(Q_{ID},P_{Pub})^r\cdot\hat{e}(xHu,rP)}{\hat{e}(xHu,rP)}=g^r$$

成立，所以如果加密运算按照规定的流程执行，解密者一定可以从密文中解密出明文消息 M 并接受该消息。

现在来分析一下 IDKIE 方案的效率。用 p 表示 pairing 运算，m 表示 G_1 上的乘法，e 表示 G_2 上的幂运算，n 表示明文长度，$|G_1|$ 表示 G_1 群上一个点的二进制表示长度。则 IDKIE

方案加密过程需要的运算量为 $2m+1p+1e$，解密过程需要的运算量为 $2p+1m$。对于密钥更新过程，Server 需要 $1m$ 来计算密钥更新信息，用户需要 $3p$ 来验证密钥更新信息完整性。密钥更新信息长度为 $2|G_1|$，密文长度为 $2n+2|G_1|$。

可以证明，IDKIE 方案在自适应选择密文攻击下是语义安全的，其详细证明可参阅文献[33]。

15.2.2 基于身份的密钥隔离签名方案

本小节基于 15.2.1 小节给出的 IDKIE 方案，构造了一个基于身份的密钥隔离签名方案(Identity-based Key Insulated Signature Scheme，IDKIS)。该签名方案的基本模型与上面给出的 IDKIE 相同，用户的私钥被拆分成两部分，分别由 User 和 Server 持有，在获得了当前时间段的密钥更新信息后，User 可以独自完成签名运算。攻击者仅攻破 User 端只能获得当前时间段的签名能力，仅攻破 Server 端不能危及任何时间段的安全，仅获得密钥更新信息也不会有助于伪造任何时间段的签名。

定义 15.9 IDKIS 是一个由以下 6 个函数组成的六元组 IDKIS：={PGen，KGen，UpdGen，UserUpd，Sign，Verify}。

(1) PGen：公开参数生成函数。输入安全参数 1^k，输出系统公开参数 params 和主密钥(Master key)s。

(2) KGen：用户私钥初始生成函数。输入用户身份 ID 和系统公开参数 params，输出用户的 Home key$\{Sk_{ID,U},Sk_{ID,S}\}$。

(3) UpdGen：密钥更新信息生成函数。输入 Server 的 Home key $Sk_{ID,S}$，时间段 t 和公开参数 params，输出 t 时间段的密钥更新信息 $Ku_{ID,t}$。

(4) UserUpd：用户密钥更新函数。输入 User 的 Home key $Sk_{ID,U}$，时间段 t，密钥更新信息 $Ku_{ID,t}$ 和公开参数 params，输出用户在 t 时间段的临时私钥 $Tk_{ID,t}$ 或者输出错误标记 $\perp$。如果成功生成 $Tk_{ID,t}$，则立即删除上个时间段的临时私钥。

(5) Sign：签名函数。输入时间段 t，t 时间段的临时私钥 $Tk_{ID,t}$，消息 M 和公开参数 params，输出对消息 M 的签名 $<t,\sigma>$。

(6) Verify：签名验证函数。输入消息 M，签名者身份信息 ID，签名结果 $<t,\sigma>$ 和公开参数 params，输出签名验证结果。如果验证通过，接受该签名；否则，拒绝。

攻击者的能力与 15.2.1 小节定义的 IDKIE 方案攻击者能力是相同的，可以询问任意用户的 Home key 和任意时间段的临时私钥。攻击者最终输出一个伪造的签名结果。

IDKIS 方案攻击者 A 可以访问多项式次的下列 Oracle。

(1) Home key Oracle O_{Hk}：与 15.2.1 小节相同。

(2) Temp key Oracle O_{Tk}：与 15.2.1 小节相同。

(3) Sign Oracle O_{sgn}：输入用户身份信息 ID、时间段 t 和消息 M，输出使用相应的临时私钥 $Tk_{ID,t}$ 对 M 的签名结果 σ。

最后，攻击者 A 输出伪造的用户 ID^* 在时间段 t^* 对消息 M^* 的签名结果 σ^*。

类似于 15.2.1 小节，这里也引入 Q-exposed 和(ID,t,Q)-compromised 的概念。称攻击者 A 最终赢得与挑战者的游戏，如果最终输出的结果 $<ID^*,t^*,M^*,\sigma^*>$ 是合法的签名结果，并且满足：该签名方案不是(ID^*,t^*,Q)-compromised；A 没有就(ID^*,t^*,M^*)询问

过 O_{Sgn}。攻击者 A 的优势定义为成功输出一个伪造签名的概率，这个概率取决于挑战者和攻击者 A 在游戏中选择的随机数。

定义 15.10 称 IDKIS 方案在选择消息攻击下是存在性非伪造的(IDKIS-UF-CMA)，如果对任意概率多项式时间攻击者 A，在上述游戏中的优势 $\mathrm{Adv}_A(k)$ 都是可忽略的。

基于双线性映射构造的 IDKIS 方案包含 6 个算法，具体过程如下。

(1) PGen：输入安全参数 1^k，PKG 生成 BDHP 参数$(q,G_1,G_2,\hat{e})$，选择 G_1 上的任意生成元 $P\in G_1$，随机选择 $s\in_R Z_q^*$，计算 $P_{\mathrm{Pub}}=\mathrm{s}\mathrm{P}$。选择 Hash 函数 $H_1:\{0,1\}^*\rightarrow G_1$，$H_m:\{0,1\}^*\rightarrow G_1$ 和 $H_u:\{0,1\}^*\rightarrow G_1$，其中 n 是明文长度。系统公开参数 params$=\{G_1,G_2,\hat{e},P,P_{\mathrm{Pub}},H_1,H_m,H_u\}$，系统主密钥为 s。

(2) KGen：输入主密钥 s，系统公开参数 params 和用户身份信息 ID，计算 $Q_{\mathrm{ID}}=H_1(\mathrm{ID})\in G_1$，$d_{\mathrm{ID}}=sQ_{\mathrm{ID}}\in G_1$。随机选择 $\mathrm{Sk}_{\mathrm{ID,U}}\in_R G_1$，令 $\mathrm{Sk}_{\mathrm{ID,S}}=d_{\mathrm{ID}}-\mathrm{Sk}_{\mathrm{ID,U}}$，输出 $\{\mathrm{Sk}_{\mathrm{ID,U}},\mathrm{Sk}_{\mathrm{ID,S}}\}$。

(3) UpdGen：输入 Server 的 Home key $\mathrm{Sk}_{\mathrm{ID,S}}$，时间段 t 和系统公开参数 params，Server 计算 $\mathrm{Hu}=H_u(t)$，随机选择 $x\in_R Z_q^*$，计算 $X=\mathrm{Sk}_{\mathrm{ID,S}}+x\mathrm{Hu}\in G_1$ 和 $Y=x\mathrm{P}\in G_1$，返回 $\mathrm{Ku}_{\mathrm{ID},t}=<X,Y>$。

(4) UserUpd：输入 User 的 Home key $\mathrm{Sk}_{\mathrm{ID,U}}$、时间段 t、密钥更新信息 $\mathrm{Ku}_{\mathrm{ID},t}=<X,Y>$和系统公开参数 params，User 首先检查等式 $\hat{e}(\mathrm{Sk}_{\mathrm{ID,U}}+X,P)=\hat{e}(Q_{\mathrm{ID}},P_{\mathrm{Pub}})\cdot\hat{e}(\mathrm{Hu}(t),Y)$是否成立。如果不成立，返回错误标记$\perp$；否则，令临时私钥 $\mathrm{Tk}_{\mathrm{ID},t}=<\mathrm{Sk}_{\mathrm{ID,U}}+X,Y>$。

(5) Sign：输入时间段 t，临时私钥 $\mathrm{Tk}_{\mathrm{ID},t}=<X_{\mathrm{ID},t},Y_{\mathrm{ID},t}>$、消息 M 和系统公开参数 params，随机选取 $r\in_R Z_q^*$，计算 $U=X_{\mathrm{ID},t}+rH_m(M)$，$V=Y_{\mathrm{ID},t}$，$W=rP$，输出签名结果$<t,\sigma>$，其中 $\sigma=<U,V,W>$。

(6) Verify：输入签名结果$<t,\sigma>$，其中 $\sigma=<U,V,W>$、用户身份信息 ID、消息 M 和系统公开参数 params，验证是否有等式 $\hat{e}(U,P)=\hat{e}(Q_{\mathrm{ID}},P_{\mathrm{Pub}})\cdot\hat{e}(H_u(t),V)\cdot\hat{e}(H_m(M),W)$成立。如果等式成立则接受该签名结果；否则，拒绝。

IDKIS 方案可以看做是“两层”的 SOK-IBS 原始签名方案[10]：临时私钥可以看做是使用 SOK-IBS 原始方案对“消息”t 的签名结果，以此作为私钥，再次使用 SOK-IBS 原始方案对“消息”M 签名就得到 IDKIS 方案的签名结果。

现在来分析一下 IDKIS 方案的效率。用 p 表示 pairing 运算，m 表示 G_1 上的乘法，e 表示 G_2 上的幂运算，$|G_1|$表示 G_1 群上一个点的二进制表示长度。则 IDKIS 方案签名过程需要的运算量为 $2m$，验证过程需要的运算量为 $4p$。对于密钥更新过程，Server 需要 $1m$ 来计算密钥更新信息，用户需要 $3p$ 来验证密钥更新信息完整性。签名长度为 $3|G_1|$。

可以证明，IDKIS 在选择消息攻击下是存在性非伪造的，详细证明可参阅文献[33]。

15.2.3 可抵抗主动攻击者的 IDKIE 方案

传统的密钥隔离方案都没有对密钥更新信息采取保护措施。15.2.1 小节的 IDKIE 方案虽然可以对密钥更新信息进行合法性检查，但没有对密钥更新信息的机密性进行保护。当然，通过分析可知，攻击者仅获得密钥更新信息不会对用户私钥安全造成危险。然而，攻击者如果侵入 User 设备并获得其中的部分私钥后，还能够一直获得密钥更新信息，那么就可以利用部分私钥和密钥更新信息获得用户以后所有时间段的临时私钥！这与密钥隔离密

码方案的初衷显然是相悖的。密钥隔离密码方案的目的就是使得攻击者即使能够攻破 User 端若干次，也只能获得攻击发生时的临时私钥。

传统密钥隔离密码方案之所以很少考虑信道安全问题，是因为这些方案大都假设在其应用环境中 User 设备（如笔记本电脑）和 Server 设备（如家中的台式机）间的密钥更新是面对面进行的（如通过数据线连接），而且由用户亲自完成。在这种情况下，密钥更新信道的安全通常是能够得到保证的。因为攻击者虽然可能较容易地窃取 User 设备并获得当时的临时私钥和 User 端的部分私钥，却很难再使用 Server 设备更新私钥。然而，如果把密钥隔离密码方案应用到网络环境中，就必须考虑密钥更新的信道安全问题。因为此时 User 和 Server 可能是两个相距甚远的计算机，密钥更新只能通过网络连接进行，而攻击者可能很容易地截获网络中传输的密钥更新信息。

密钥更新信道安全问题初看上去似乎很容易解决：使用 TLS/SSL 协议在 User 和 Server 间建立一个安全信道，认证并秘密地传送密钥更新信息。然而这种办法虽然能够抵抗被动攻击者（即只被动监听信道消息的攻击者）的攻击，对主动攻击者（即除监听外还主动发送、删除、修改信道中的消息）却仍然是脆弱的。因为当攻击者侵入了 User 端设备后，除了可以获得 User 的 Home key 和临时私钥外，还可以获得 User 用于建立安全信道的认证密钥，这样攻击者就可以使用该密钥假冒 User 向 Server 请求密钥更新，从而不断地获得用户的临时私钥。当然如果用户能及时发现攻击者假冒 User，就可以通过更换密钥的方式阻止攻击者继续获得临时私钥。但通常用户只有在灾难之后才察觉到私钥泄露。因此，需要考虑的是如何在主动攻击者控制的信道上，安全地实现密钥隔离密码方案的密钥更新过程，防止攻击者通过获得 User 端部分私钥和密钥更新信息长期地获得用户的私钥。

使用防窜扰密码设备可以防止攻击者获得部分私钥和认证私钥，但就目前的实际情况看，防窜扰密码设备仍不普及，而且成本较高，对普通应用来说并不实际。因此，这里必须假设攻击者在侵入 User 设备后可以获得 User 的所有私钥。

首先必须接受的事实是，在攻击者掌握了 User 所有私钥后，除非采用离线认证方式，否则，攻击者总能假冒 User 获得密钥更新信息。为此，退而求其次：使 User 能够在较短的时间内发现这种攻击行为，或者使攻击者很快就失去对 User 的部分私钥的掌握，从而避免攻击者长期获得用户私钥。

基本思路是使用一个带密钥确认的认证密钥协商协议（Authenticated Key agreement protocol with key Confirmation，AKC），在 User 和 Server 之间建立一个安全信道用于传输密钥更新信息，并且作为一个安全的前摄更换协议（Securely Proactive Refreshment protocol，SPR）随机更换双方的 Home key，而同时保持用户的完整私钥不变。

在这个方案中，User 向 Server 请求密钥更新信息时会调用 SPR 协议认证协商一个用于安全传输密钥更新信息的会话密钥，同时自动更换双方的 Home key。这样，即使攻击者已经掌握了 User 当前的所有私钥（Home key 和 AKC 私钥），如果攻击者只是被动监听信道，则基于 AKC 协议安全性，攻击者不能获得信道上传输的任何密钥更新信息，对前摄更换后 User 的 Home key 也一无所知；如果攻击者主动假冒 User 向 Server 请求密钥更新信息，则 Server 必然自动调用 SPR 协议更换自己的 Home key，这样，之后 User 向 Server 请求密钥更新时就会发现密钥更新信息不合法，进而察觉到攻击者的存在。

为了达到这一目的，要求 AKC 协议须具有以下性质。

(1) 已知密钥安全(Known-key security)。某次密钥协商生成的会话密钥的泄露不会危及其他会话密钥的安全。

(2) 前向机密性(Forward secrecy)。如果认证双方用于密钥协商的长期私钥泄露,之前协商产生的会话密钥的安全不会受到影响。

(3) 抗已知私钥假冒(Key-compromise impersonate resilience)。攻击者即使已知实体 A 用于密钥协商的长期私钥,也不能向 A 冒充其他实体进行密钥协商。

(4) 抗未知密钥共享(Unknown key-share resilience)。攻击者不能欺骗实体 A 与实体 C 建立一个会话密钥,而让 A 以为是在与实体 B 进行密钥协商。

1. AR-IDKIE 模型

基于上述思路,在 15.2.1 小节 IDKIE 方案的基础上,给出了一个可抵抗主动攻击者的密钥隔离加密方案(Active-adversary-Resistant IDKIE, AR-IDKIE)。AR-IDKIE 方案同 IDKIE 方案基本类似:用户的私钥被拆分成两部分,分别由 User 和 Server 存储,系统整个生命周期被划分成连续的时间段,在每个时间段结束时,User 向 Server 请求密钥更新。但与 IDKIE 方案不同的是,在 AR-IDKIE 方案中,User 与 Server 间的交互使用 SPR 协议安全地进行。为了执行 SPR 协议,User 和 Server 除了持有各自的 Home key 外,还持有用于 AKC 协议的认证密钥(Authentication key),User 和 Server 的认证私钥分别记为 $Ak_{ID,U}$ 和 $Ak_{ID,S}$。每次成功执行 SPR 协议后,User 和 Server 的 Home key 都会自动更换,但用户的完整私钥保持不变。称 Home key 更换的次数为更换阶段,记为 τ,用户初始注册时,$\tau=0$。

在 AR-IDKIE 方案中,假设 User 端以一种只读的方式或可检验完整性的方式存放系统的公开参数,Server 的认证公钥及协议代码。这个假设是可行的,因为上述信息在整个生命期内保持不变,可以以只读形式存储。另外,这个假设也是必需的,因为如果上述信息可以被攻击者修改,那么当攻击者离开 User 设备时,仍然可以利用修改过的协议代码或系统参数继续控制用户设备的行为。实际上,这个假设在目前 PKI 系统中也是必需的,因为如果攻击者可以替换用户系统中存储的 CA 证书,那么攻击者可以利用用户对 CA 的绝对信任进行各种攻击。在 PKI 中,总是假设有多种方法确保用户系统中的 CA 证书不会被替换(如通过各种公开渠道发布 CA 证书),在 AR-IDKIE 方案中也是同样的。

定义 15.11 AR-IDKIE 是一个由以下 7 个函数组成的七元组 AR-IDKIE:={PGen, KGen, SPR, UpdGen, UserUpd, Enc, Dec}。

(1) PGen:公开参数生成函数。输入安全参数 1^k,输出系统公开参数 params 和主密钥(Master key)s。

(2) KGen:用户私钥初始生成函数。输入用户身份 ID 和系统公开参数 params,输出用户的 Home key $\{Sk^0_{ID,U}, Sk^0_{ID,S}\}$ 和认证密钥 $\{Ak_{ID,U}, Ak_{ID,S}\}$。

(3) SPR:安全前摄更换协议。输入 User 和 Server 的认证密钥 $\{Ak_{ID,U}, Ak_{ID,S}\}$,双方的 Home key $\{Sk^\tau_{ID,U}, Sk^\tau_{ID,S}\}$,以及系统公开参数 params,输出会话密钥 $Sesk^\tau_{ID}$ 和新的 Home key $\{Sk^{\tau+1}_{ID,U}, Sk^{\tau+1}_{ID,S}\}$,并删除旧的 Home key $\{Sk^\tau_{ID,U}, Sk^\tau_{ID,S}\}$。

(4) UpdGen:密钥更新信息生成函数。输入 Server 的 Home key $Sk^\tau_{ID,S}$,时间段 t 和公开参数 params,输出 t 时间段的密钥更新信息 $Ku^\tau_{ID,t}$。

(5) UserUpd:用户密钥更新函数。输入 User 的 Home key $Sk^\tau_{ID,U}$、时间段 t、密钥更新

信息 $Ku_{ID,t}^{\tau}$ 和公开参数 params，输出用户在 t 时间段的临时私钥 $Tk_{ID,t}$ 或者输出错误标记 ⊥。如果成功生成 $Tk_{ID,t}$，则立即删除上个时间段的临时私钥。

(6) Enc：加密函数。输入用户身份信息 ID、时间段 t、明文 M 和公开参数 params，输出密文 $\langle t,C\rangle$。

(7) Dec：解密函数。输入密文 $\langle t,C\rangle$、t 时间段的临时私钥 $Tk_{ID,t}$ 和公开参数 params，输出明文 M 或者输出错误标记 ⊥。

每次 User 请求密钥更新时，双方使用 SPR 协议认证对方身份并协商一个会话密钥，然后，Server 生成密钥更新信息 $Ku_{ID,t}^{\tau}$ 并使用会话密钥安全地传送给 User，User 在验证密钥更新信息正确后，生成新的临时私钥 $Tk_{ID,t}$，并删除上一时间段的临时私钥。之后，双方均从会话密钥计算一个密钥更换参考值，根据这个值更换自己的 Home key，并安全删除旧的 Home key。

除了具有 IDKIE 方案的安全性以外，AR-IDKIE 方案还可以使已经获得 User 端 Home key 和认证私钥的攻击者在一个时间段内或者被发现，或者丧失对 User 端 Home key 的掌握。这种安全性依赖于 AKC 协议的支持，在前面给出了对 AKC 协议的安全要求，相关安全性定义已经在文献[51]～[53]中给出，文献[22]和[25]还给出了基于身份的 AKC 协议的安全性定义。下面在 AKC 协议安全的假设下，定义 AR-IDKIE 方案的安全性。

AR-IDKIE 方案的目标是：攻击者多次攻破 User 也只能获得有限时间段的解密能力；仅获得 Server 端的 Home key 不会危及任何时间段的安全。攻击者仅获得密钥更新信息或者密钥更换参考值不会危及任何时间段的私钥安全。

必须指出，如果攻击者获得了 Server 的 Home key 和认证私钥，就可以向 User 假冒 Server 而不会被 User 发现。而且如果 Server 只被动响应请求，而不主动向 User 发送密钥更新信息，那么 Server 也很难发现自己被攻击者假冒。这样攻击者就可以与 User 同步更换 Home key，并伺机再窃取 User 端的 Home key，最终获得用户的完整私钥。这种攻击行为仅靠密码算法无法抵抗，但 Server 通常都是安全性较高的密码设备或系统部件，因此在这里，假设 Server 总是安全的，当然，仅获得 Server 端的 Home key 不会危及任何时间段的安全。

令 A 是 AR-IDKIE 方案的一个概率多项式时间攻击者，为了模仿实际环境中攻击者的行为，允许 A 在与挑战者的游戏中访问下面的各种 Oracle。

(1) Home key Oracle O_{Hk}：输入用户的身份信息 ID，$P\in\{U,S\}$ 以及 Home key 更换阶段 τ，输出对应的 Home key $Sk_{ID,P}^{\tau}$。

(2) Temp key Oracle O_{Tk}：输入用户身份信息 ID 和时间段 t，输出对应的临时私钥 $Tk_{ID,t}$。

(3) Key refreshment Oracle O_{Kr}：输入用户身份信息 ID 以及 Home key 更换阶段 τ，输出 Home key 更换参考值 R_{ID}^{τ}。

(4) Dec Oracle O_{Dec}：输入用户身份信息 ID 和密文 $\langle t,C\rangle$，输出使用临时私钥 $Tk_{ID,t}$ 对 C 的解密结果。

(5) L-R Oracle O_{LR}：输入用户身份信息 ID^*、时间段 t^* 和两个等长的明文 m_0、m_1，输出挑战密文 $C^*=\mathrm{Enc}(ID^*,t^*,m_b)$，其中 $b\in\{0,1\}$ 是事先选定的。攻击者猜测 b 的值。

同 15.2.1 小节的 IDKIE 方案安全性定义相比，AR-IDKIE 方案攻击者增加了访问

Key Refreshment Oracle 的能力，可以获得 SPR 协议产生的 Home key 更换参考值。增加这个 Oracle 是为了模拟实际环境中攻击者可能在 SPR 协议运行时攻破 User 设备，从而获得 SPR 协议的中间结果。另外同 IDKIE 方案安全性定义相同，不直接提供密钥更新信息 Oracle，而是通过 Home key Oracle 和 Temp key Oracle 来模拟攻击者获取密钥更新信息的行为。

类似于 15.2.1 小节，为了方便描述，把 O_{Hk}、O_{Tk}、O_{Kr} 3 种 Oracle 统一表示成 O_{Sec}，并使用不同的字符串区分。

攻击者可以任意询问上述 Oracle 多项式次(L-R Oracle 仅询问一次)，但为了模拟实际情况，要求攻击者必须遵守"删除原则"，即攻击者在询问过更换阶段 τ 时的 Home key 后，不能再询问之前阶段的 Home key 和密钥更换参考值，因为在实际实现时，这些数据应该都已被安全删除。该原则形式化地描述如下。

(1) $O_{\mathrm{Sec}}(\mathrm{Hk},\mathrm{ID},P,\tau)$必须在 $O_{\mathrm{Sec}}(\mathrm{Hk},\mathrm{ID},P,\tau')$之前询问，如果 $\tau'>\tau$。

(2) $O_{\mathrm{Sec}}(\mathrm{Hk},\mathrm{ID},P,\tau)$必须在 $O_{\mathrm{Sec}}(\mathrm{Kr},\mathrm{ID},P,\tau')$之前询问，如果 $\tau'>\tau$。

对于一个 O_{Sec} 询问集合 Q，则称 Home key $\mathrm{Sk}_{\mathrm{ID},P}^{\tau}$ 是 Q-exposed 的，如果符合以下某一条件：

(1) $O_{\mathrm{Sec}}(\mathrm{Hk},\mathrm{ID},P,\tau)\in Q$。

(2) $\tau>0$，$O_{\mathrm{Sec}}(\mathrm{Kr},\mathrm{ID},\tau-1)\in Q$ 且 $\mathrm{Sk}_{\mathrm{ID},P}^{\tau-1}$ 是 Q-exposed 的。

(3) $O_{\mathrm{Sec}}(\mathrm{Kr},\mathrm{ID},\tau)\in Q$ 且 $\mathrm{Sk}_{\mathrm{ID},P}^{\tau+1}$ 是 Q-exposed 的。

称临时私钥 $\mathrm{Tk}_{\mathrm{ID},t}$ 是 Q-exposed 的，如果 $O_{\mathrm{Sec}}(\mathrm{Tk},\mathrm{ID},t)\in Q$。

如果临时私钥 $\mathrm{Tk}_{\mathrm{ID},t}$ 是 Q-exposed 的，或者存在一个 $\tau>0$，使得 $\mathrm{Sk}_{\mathrm{ID},\mathrm{U}}^{\tau}$ 和 $\mathrm{Sk}_{\mathrm{ID},\mathrm{S}}^{\tau}$ 都是 Q-exposed 的，则称该 AR-IDKIE 方案是(ID,t,Q)-compromised 的。

在游戏中，如果攻击者最终在 L-R Oracle 询问中输出了对 b 正确的猜测 b'，使得 $b=b'$，并且满足以下条件：该 IDKIE 方案不是(ID^*,t^*,Q)-compromised 的；假设 O_{LR} 返回的密文为 C^*，攻击者没有向解密 Oracle O_{Dec} 询问过 $O_{\mathrm{Dec}}(ID^*,t^*,C^*)$，则称攻击者赢得了与挑战者的游戏。攻击者获胜的优势定义为 $\mathrm{Adv}_A(k)=|\Pr[b=b']-1/2|$。

定义 15.12 称 AR-IDKIE 方案在自适应选择密文攻击下是语义安全的(AR-IDKIE-CCA)，如果对任意概率多项式时间攻击者 A，在上述游戏中的优势 $\mathrm{Adv}_A(k)$都是可忽略的。

2. 基于双线性映射的 AR-IDKIE 方案

这里基于双线性映射构造了一个 AR-IDKIE 方案。该方案可以使用任何满足前面所述要求的 AKC 协议。为了能够在 ID-PKC 中提供一个完整的 AR-IDKIE 方案，使用了文献[25]中给出的基于身份的 AKC 协议。

文献[25]中的 AKC 协议共有 3 个函数，具体过程如下。

(1) Setup：对输入的安全参数 $k\in Z^+$，BDH 参数生成器 G 生成一个双线性群 $G_p=\{q,G_1,G_2,\hat{e}\}$。令 h 为 q 在 G_1 群上的余因子(Co-factor)(即 G_1 的椭圆曲线基群的阶为 qh)，P 为G_1 的生成元。选择一个随机的主密钥 $s\in Z_q^*$ 和 4 个单向函数 $H:\{0,1\}^*\rightarrow G_1$，$\pi:G_1\times G_1\rightarrow Z_q^*$，$H_1,H_2:G_2\rightarrow\{0,1\}^n$，其中 n 是会话密钥的长度，以及一个安全的 MAC 函数 $\mathrm{MAC}_{(.)}(\cdot)$。系统公开参数为$\langle q,h,P,G_1,G_2,\hat{e},H,\pi,H_1,H_2,\mathrm{MAC}\rangle$，主密钥为 s。

(2) Extract：对一个给定的身份 $ID\in\{0,1\}^*$，计算 $Q_{ID}=H(ID)\in G_1$，则该用户的私钥为 $d_{ID}=sQ_{ID}\in G_1$。

(3) Exchange：假定该协议的两个参与方 A 和 B 的身份分别为 ID_A 和 ID_B，私钥分别为 d_A 和 d_B，则交互过程如下。

① A 选择随机数 $x\in_R Z_q^*$，计算 $R_A=xQ_A$，并发送 R_A 给 B。

② B 选择随机数 $y\in_R Z_q^*$，计算 $R_B=yQ_B$，$s_A=\pi(R_A,R_B)$，$s_B=\pi(R_B,R_A)$，以及共享秘密 $sk_{IDAK}=\hat{e}(Q_A,Q_B)^{(x+s_A)(y+s_B)hs}=\hat{e}(s_AQ_A+R_A,(y+s_B)hd_B)$。并计算 $K_1=H_1(sk_{IDAK})$，$K_2=H_2(sk_{IDAK})$。最后 B 发送 R_B 和 $MAC_{K_2}(ID_B,ID_A,R_B,R_A)$ 给 A。

③ A 收到 B 的消息后，计算 $s_A=\pi(R_A,R_B)$，$s_B=\pi(R_B,R_A)$，以及共享秘密 $sk_{IDAK}=\hat{e}(Q_A,Q_B)^{(x+s_A)(y+s_B)hs}=\hat{e}(s_BQ_B+R_B,(x+s_A)hd_A)$。并计算 $K_1=H_1(sk_{IDAK})$，$K_2=H_2(sk_{IDAK})$。最后 A 发送 $MAC_{K_2}(ID_A,ID_B,R_A,R_B)$ 给 B。

在 15.2.1 小节的 IDKIE 方案基础上，通过增加 SPR 协议得到了可前摄更换 Home key 的 AR-IDKIE 方案，加、解密过程和密文空间与 15.2.1 小节的 IDKIE 方案完全相同。AR-IDKIE 方案共包括 7 个函数，具体过程如下。

(1) PGen：输入安全参数 1^k，PKG 生成 BDH 参数$(q,G_1,G_2,\hat{e})$，选择 G_1 上的任意生成元 $P\in G_1$，随机选择 $s\in_R Z_q^*$，计算 $P_{Pub}=sP$。选择 Hash 函数 $H_1:\{0,1\}^*\to G_1$，$H_2:G_2\to\{0,1\}^n$，$H_3:\{0,1\}^n\times\{0,1\}^n\to Z_q^*$，$H_4:\{0,1\}^n\to\{0,1\}^n$，$H_u:\{0,1\}^*\to G_1$ 和 $H_r:G_2\to G_1$，其中 n 是明文长度。系统公开参数 $params=\{G_1,G_2,\hat{e},P,P_{Pub},H_1,H_2,H_3,H_4,H_u,H_r\}$，系统主密钥为 s。

(2) KGen：输入主密钥 s，系统公开参数 params 和用户身份信息 ID，计算 $Q_{ID}=H_1(ID)\in G_1$，$d_{ID}=sQ_{ID}\in G_1$。随机选择 $Sk_{ID,U}^0\in_R G_1$，令 $Sk_{ID,S}^0=d_{ID}-Sk_{ID,U}^0$，输出 $\{Sk_{ID,U}^0,Sk_{ID,S}^0\}$。

(3) SPR：User 和 Server 使用各自的认证密钥 $Ak_{ID,U}$ 和 $Ak_{ID,S}$ 运行 AKC 协议互相认证身份，并得到一个共享秘密 SS_{IDAK}。双方计算 Home key 更换参考值 $R_{ID}^{\tau}=H_r(SS_{IDAK})\in G_1$ 和会话密钥 $Sesk_{ID}=H_1(SS_{IDAK})$。User 更换其 Home key 为 $Sk_{ID,U}^{\tau+1}=Sk_{ID,U}^{\tau}+R_{ID}^{\tau}$，Server 更换其 Home key 为 $Sk_{ID,S}^{\tau+1}=Sk_{ID,S}^{\tau}-R_{ID}^{\tau}$，双方删除各自旧的 Home key $Sk_{ID,U}^{\tau}$ 和 $Sk_{ID,S}^{\tau}$。

(4) UpdGen：输入 Server 的 Home key $Sk_{ID,S}^{\tau}$、时间段 t 和系统公开参数 params，Server 计算 $Hu=H_u(t)$，随机选择 $x\in_R Z_q^*$，计算 $X=Sk_{ID,S}^{\tau}+xHu\in G_1$ 和 $Y=xP\in G_1$，返回 $Ku_{ID,t}^{\tau}=\langle X,Y\rangle$。

(5) UserUpd：输入 User 的 Home key $Sk_{ID,U}^{\tau}$、时间段 t、密钥更新信息 $Ku_{ID,t}^{\tau}=\langle X,Y\rangle$ 和系统公开参数 params，User 首先检查等式 $\hat{e}(Sk_{ID,U}^{\tau}+X,P)=\hat{e}(Q_{ID},P_{Pub})\cdot\hat{e}(Hu(t),Y)$ 是否成立。如果不成立，返回错误标记⊥；否则令临时密钥 $Tk_{ID,t}=\langle Sk_{ID,U}^{\tau}+X,Y\rangle$。

(6) Enc：输入用户身份信息 ID、时间段 t、明文 M 和系统公开参数 params，计算 $Q_{ID}=H_1(ID)$，随机选择 $\sigma\in_R\{0,1\}^n$，计算 $r=H_3(\sigma,M)$，$Hu=H_u(t)$ 和 $g=\hat{e}(Q_{ID},P_{Pub})$。令 $C=\langle rHu,rP,\sigma\oplus H_2(g^r),M\oplus H_4(\sigma)\rangle$。返回 $\langle t,C\rangle$。

(7) Dec：输入密文 $\langle t,C\rangle$ 其中 $C=\langle T,U,V,W\rangle$，临时密钥 $Tk_{ID,t}=\langle X_{ID,t},Y_{ID,t}\rangle$

和系统公开参数 params，首先计算 $\sigma=V\oplus H_2\left(\frac{\hat{e}(X_{\mathrm{ID},t},U)}{\hat{e}(T,Y_{\mathrm{ID},t})}\right)$，$M=H_4(\sigma)$，其次令 $r=H_3(\sigma,M)$，检查是否 $U=rP$。如果不等，返回错误标记⊥；否则，返回 M。

AR-IDKIE 方案的加、解密效率与 5.2.1 小节的 IDKIE 方案相同。用 p 表示 pairing 运算，m 表示 G_1 上的乘法，e 表示 G_2 上的幂运算。则 AR-IDKIE 方案加密过程需要的运算量为 $2m+1p+1e$，解密过程需要的运算量为 $2p+1m$。SPR 协议需要在 User 和 Server 间进行两轮交互，根据文献[25]中的分析结果，在使用预计算的情况下，运行 SPR 协议需要双方各计算 $1p+1e$。对于密钥更新过程，Server 需要 $1m$ 来计算密钥更新信息，用户需要 $3p$ 来验证密钥更新信息完整性。

对 AR-IDKIE 方案的安全性分析分两步进行，首先分析方案中 SPR 协议的安全性，然后在 SPR 协议安全性的基础上，分析 AR-IDKIE 整体方案的安全性。详细分析可参阅文献[33]。

15.3 ID-PKI 密钥管理方案

ID-PKI 在许多方面比传统 PKI 更具优势，如结构简单、使用方便等，但在拥有这些优势的同时，ID-PKI 也还有若干问题尚待解决，最突出的就是 ID-PKI 的密钥管理问题。该问题主要包括用户私钥分发、密钥撤销和密钥托管 3 个方面。目前针对这些问题已有若干解决方案。但这些方案大多只解决了部分问题，目前尚没有一个针对上述 3 个问题的完整解决方案，而且由于所基于的底层密码算法不同，不同方案也很难配合使用，有的方案甚至还存在一定的安全缺陷。针对这种情况，本节利用 15.2 节中提出的基于身份的密钥隔离密码方案，给出了针对 ID-PKI 密钥管理问题的完整解决方案。

15.3.1 研究背景和相关研究进展

ID-PKI 中以用户身份信息作为公钥，密钥中心(PKG)使用系统主密钥为每个用户生成私钥，这就导致 ID-PKI 密钥管理与传统基于证书的 PKI 密钥管理有很大不同，相对而言，前者更复杂，解决更困难。

首先，由于 ID-PKI 中所有用户私钥都由 PKG 产生，因此必然要求 PKG 把私钥安全地分发到用户手中，而在传统 PKI 中，用户私钥通常由用户自己生成，因此不需要特别考虑密钥分发问题。当然，如果单纯考虑 ID-PKI 中私钥分发的安全性，那么该问题很容易解决，比如可以离线认证用户身份并面对面地分发私钥，但这种方式只适于规模较小，而且密钥分发周期较长的应用环境。当大规模应用或者密钥有效期比较短时，这种方式效率太低，难以实用。如何在保证用户私钥传输安全性的同时，尽可能提高效率，并减少用户参与是密钥分发方案需要着重研究的内容。

其次，由于 ID-PKI 中用户公钥与身份是一一绑定的，传统 PKI 中的各种密钥撤销方案在 ID-PKI 环境中并不适用，主要表现在以下两个方面。

(1) 传统的密钥撤销方案如 CRL、OCSP[8,9] 等要求依赖方在使用对方公钥前，通过查询证书状态的方式判断对方密钥是否撤销。但在 ID-PKI 中，由于没有了证书这一载体，并且用户身份表示方法可能不确定，依赖方很难精确地查询到对方的密钥撤销状态。而且即

使能够根据用户身份信息精确地查询到密钥撤销状态，这种查询方式也违背了 ID-PKC 的初衷[1]，即依赖方不必事先向第三方查询就可以使用对方的公钥加密消息或验证签名。

(2) 传统 PKI 中，用户撤销原有密钥后，可以更换一对新公、私钥，再由 CA 重新颁发一个证书。但在 ID-PKI 中，公钥与用户身份一一对应，用户不可能随便更换自己的身份。因此，ID-PKI 中的密钥撤销方案应对依赖方尽可能的透明，使得依赖方在使用对方公钥加密消息或验证签名前不必先查询其撤销状态。另外，密钥撤销方案还应该允许在撤销并更换用户私钥的同时，保持用户的公钥即身份不需改变。

最后，密钥托管是 ID-PKC 天生存在的问题：PKG 为所有用户生成私钥，自然也就掌握了所有用户私钥。密钥托管虽然在某些情况下是有益的(如政府部门需要检查犯罪分子通信内容时)，在更多情况下却是应该避免的，如密钥托管导致了不可否认性服务无法实现，而不可否认性是电子商务中不可缺少的一类安全服务。因此，如何恰当地解决密钥托管问题也是设计 ID-PKI 系统时需要着重考虑的。

1. 密钥撤销

Boneh 和 Franklin 在文献[5]中除了给出 IBE 方案外，还给出了一个密钥撤销方案(以下简称 BF 方案)。在该方案中，用户的公钥中除了包含其身份信息外，还包含一个密钥有效期，如“Alice ‖ 2009-12”。这样，在有效期过后，该密钥被自动撤销。BF 方案非常简单，不会给用户和 PKG 增加任何计算量和通信量。但该方案只能被动撤销，不能在有效期结束前强制撤销用户密钥。这样，如果用户私钥在其有效期开始时就被攻击者窃取，即使用户能够立即发现私钥泄露，也只能等待该密钥过期后自动撤销，由此可能造成较大的损失。为了降低密钥泄露造成的损失，该方案必须使密钥的有效期非常短，但这就使得用户必须频繁地向 PKG 申请私钥，给用户和 PKG 都造成很大负担，降低了系统的效率。

在文献[54]中，Boneh 等人提出了使用安全中介部件(Security Mediator，SEM)实现密钥撤销功能的思想，并给出了一个中介 RSA(mediated RSA)算法，之后又提出了基于身份的中介 RSA 算法[55,56]。Libert 和 Qusquater 在 Boneh-Fraklin IBE 方案基础上构造了门限 IBE 方案，并借鉴 Boneh 等人中介 RSA 的思想，使用(2,2)门限 IBE 提出了一个基于 SEM 的密钥撤销方案[57](以下简称 LQ 方案)。上述方案的基本思想都是把用户私钥拆分成两部分，分别由用户和 SEM 掌握，用户所有的密码运算都需要与 SEM 共同完成。用户每次解密密文或签名消息时，需先向 SEM 请求协助；收到用户请求后，SEM 先检查用户的撤销状态，如果该用户密钥没有被撤销，SEM 使用自己掌握的部分私钥计算一个中间结果发送给用户，否则拒绝响应用户请求。用户根据 SEM 返回的中间结果和自己计算的中间结果得出最后的解密或签名结果。Libert 和 Quisquater 指出所有的安全门限密码方案都可以用于构造基于 SEM 的密钥撤销方案，不过他们给出的门限 IBE 方案只具有弱 CCA 语义安全性[57]，Baek 和 Zheng 对该方案又进一步改进[58]，使其加密方案达到 CCA 语义安全。

LQ 方案的优点是可以实现 SEM 对用户密钥的即时撤销，因为收到密钥撤销请求后，SEM 可以使用户立即丧失解密或签名能力。这样，LQ 方案可以提供极细粒度的密钥撤销效果。然而，为了实现这一效果，该方案在其他性能上做了较大牺牲。首先，可以看出，该方案中用户每次解密或签名都必须在线等待 SEM 的回应，这就降低了用户的解密/签名速度，并增加了用户的网络通信负担。其次，该方案使得用户不能进行离线的解密/签名运算，然而在很多场合中，离线解密/签名是必需的。另外，在一个 ID-PKI 系统中，SEM 必须处理

大量的用户解密/签名请求，当系统规模比较大使得用户解密频繁时，SEM 容易成为系统的瓶颈。

文献[59]提出了一种基于身份的多层密钥隔离加密方案(IKE)，使用户不与 PKG 交互就可以更新私钥，并借此提供一种密钥撤销手段。为了获得高安全性，该方案把私钥拆分成多层，每一层对应不同长度的时间段(如 3 个月或 1 天)，分别存放在不同的硬件设备中。这个方案在降低密钥泄露风险方面比较有效，但就密钥撤销而言仍有不足。密钥撤销根据发起方不同可以大致分为两类：用户发起的和系统发起的。由用户发起的密钥撤销很容易理解，当用户发现私钥泄露后自然需要撤销该密钥。由系统发起的密钥撤销可能不常见，却同样重要，比如当某公司职员辞职后，该公司必须立即撤销之前为该职员颁发的密钥，否则该职员辞职后仍可能获得公司内部信息。在文献[59]的方案中，如果支持系统发起的密钥撤销，则必须有至少一层的私钥由某个系统部件掌握。然而，这样该方案的非交互密钥更新特性就丧失了，而且同样存在传统密钥隔离方案的信道安全问题。

2. 密钥托管

在 BF 方案[5]中，Boneh 和 Franklin 建议使用门限密码方法把主密钥分散存放在多个 PKG 中，以保证主密钥的安全性，同时避免密钥托管。Chen 等人在文献[60]中也采用了多个 PKG 方法避免密钥托管问题，但与文献[5]的区别是，该方案中系统的完整主密钥是所有 PKG 主密钥之和，而不是采用门限方法得出的。

另一类解决密钥托管问题的方法是除了 PKG 生成的私钥外，用户自己还选择一部分私钥，这样 PKG 无法获得用户的完整私钥信息，这类方法包括 Riyami 和 Paterson 的无证书(Certificateless)密码方案[61]、Gentry 的基于证书(Certificate-based)的密码方案和 Cheng 等人在文献[62]中提出的方案等。这类方案不需要增加 PKG 的个数，但在避免密钥托管的同时，这些方案也丧失了 ID-PKC 的优势，因为在这些方案中，依赖方在加密消息或者验证签名时，必须像传统 PKI 中查找证书那样，事先查找对方的另一部分公钥。

Lee 等人提出了一种 ID-PKC 密钥分发协议[63](以下简称 Lee 方案)以避免密钥托管问题。在该方案中，用户私钥由一个 KGC(Key Generation Center)和多个 KPA(Key Privacy Authority)共同产生，这样 KGC 不能掌握用户的完整私钥。该方案为了提高用户申请私钥的效率，仅要求用户向 KGC 认证身份，而不需向 KPA 认证。但可以发现，如果 KPA 不验证请求者身份，那么 KGC 完全可以假冒用户向 KPA 索取私钥信息，最后得到用户的完整私钥，因此，该方案实际上并没有彻底解决密钥托管问题。但如果要求 KPA 也认证用户身份，那么该方案与多 PKG 的方案[5, 60]相比并没有太大优势，甚至效率更低，因为该方案需要 KGC 和所有 KPA 串行计算用户私钥。Gangishetti 等人提出了一个对 Lee 方案的改进方案[64]，提高了效率。Oh 等人借鉴 Lee 方案的密钥托管解决办法和 LQ 方案的密钥撤销思想，提出了一种试图同时解决密钥托管和密钥撤销问题的方案(以下简称 OLM 方案)[65]。但这两个方案也和 Lee 等人的方案一样都没有彻底解决密钥托管问题。关于 Lee 方案的分析还可参见文献[66,67]。

3. 密钥分发

Lee 方案[63]中的密钥分发协议使用盲因子来保证私钥信息的机密性。用户在申请私钥前，先选择一个随机数 x，计算 $X=xP$，在向 KGC 和 KPA 申请私钥时发送 X。这样只有

知道 x 的实体，即用户自己，能够从 KGC 和 KPA 返回的消息中得到完整的用户私钥。该方案的安全前提是 X 在从用户到 KGC 和 KPA 传输过程中不会被篡改，否则攻击者可以通过替换 X 获得用户私钥。为了保证 X 的完整性，可以通过离线方式或预设安全信道的方式传送 X，文献[63]没有就此问题详细讨论。

Sui 等人在文献[68]中给出了一个 ID-PKC 中匿名并保密分发用户私钥的方案（以下简称 Sui 方案）。该方案中存在一个为所有用户生成私钥的 KGC 和一个用于验证用户身份的本地注册权威（Local Registration Authority，LRA）。用户在申请私钥时，首先向 LRA 离线认证身份，并注册一个口令，LRA 把用户身份信息和口令安全地传送给 KGC。之后，用户凭口令向 KGC 认证身份，并使用盲因子保证私钥从 KGC 到用户的传输过程中的机密性。申请私钥成功后，用户和 KGC 均立即删除口令。该方案通过使用口令保证盲因子不会被攻击者替换。该方案的安全前提是 LRA 是可信的，并且 LRA 和 KGC 之间有安全信道，这与传统 PKI 是类似的。传统 PKI 中，RA 也必须是可信的，并且 RA 和 CA 之间必须有可信信道以保证用户证书申请信息不会被篡改。Sui 方案使用多个 KGC 共同产生用户私钥以避免密钥托管。

15.3.2　基于密钥隔离密码方案的 ID-PKI 密钥管理方案

利用 15.2 节提出的基于身份的密钥隔离密码方案，可以构造一个 ID-PKI 系统，并对密钥分发、密钥撤销和密钥托管问题提供完整解决方案。

实际应用环境中的密钥撤销可以分为用户发起和系统发起两种。为了支持系统发起的密钥撤销，必须有一个系统部件能够控制用户的密码运算能力，如 BF 方案[5]中的 PKG 和 LQ 方案[57]中的 SEM。容易发现，15.2 节的密钥隔离密码方案中的 Server 部件也具有这种能力。密钥隔离密码方案中的 Server 部件可以通过控制发送密钥更新信息实现对 User 解密/签名能力的控制。可以利用密钥隔离密码方案的这个特性，设计 ID-PKI 密钥管理方案。

1. 系统结构

方案中设置一个可信系统部件，称其为 SEM（Security Mediator）。SEM 持有所有用户 Server 端的 Home key，用户持有各自 User 端的 Home Key，SEM 定期更新用户的临时私钥，并通过控制密钥更新实现对用户密钥撤销状态的控制。

类似于 BF 方案，该方案通过设置多个 PKG 来解决密钥托管问题，每个 PKG 都是独立自治的，各持有一部分主密钥，超过门限数量的 PKG 可以计算出完整的用户私钥。这里使用门限密码的方法而不是像文献[60]中简单相加的方法是为了增加 PKG 的鲁棒性，使得在少量 PKG 损坏的情况下，系统仍能保持正常运行。

文献[63]指出，使用多个 PKG 造成每次申请私钥都需要用户多次认证身份，增加了用户负担。为此，借鉴 Sui 方案[68]和传统 PKI 中的方法，在系统中设置注册权威机构（RA）用于在初始注册时离线验证用户身份，并把用户的注册信息安全地传送给 PKG。与 Sui 方案和传统 PKI 类似，在该方案中 RA 也必须是可信系统部件。

整个系统的逻辑结构如图 15.2 所示。

图 15.2 中还给出了密钥分发和密钥撤销的流程，其中第①～④步是密钥分发过程，第⑤、⑥步是密钥撤销过程。

在这个系统中，假设 RA、SEM 和 PKG 都是可信的，而且这些系统部件间都已经存在

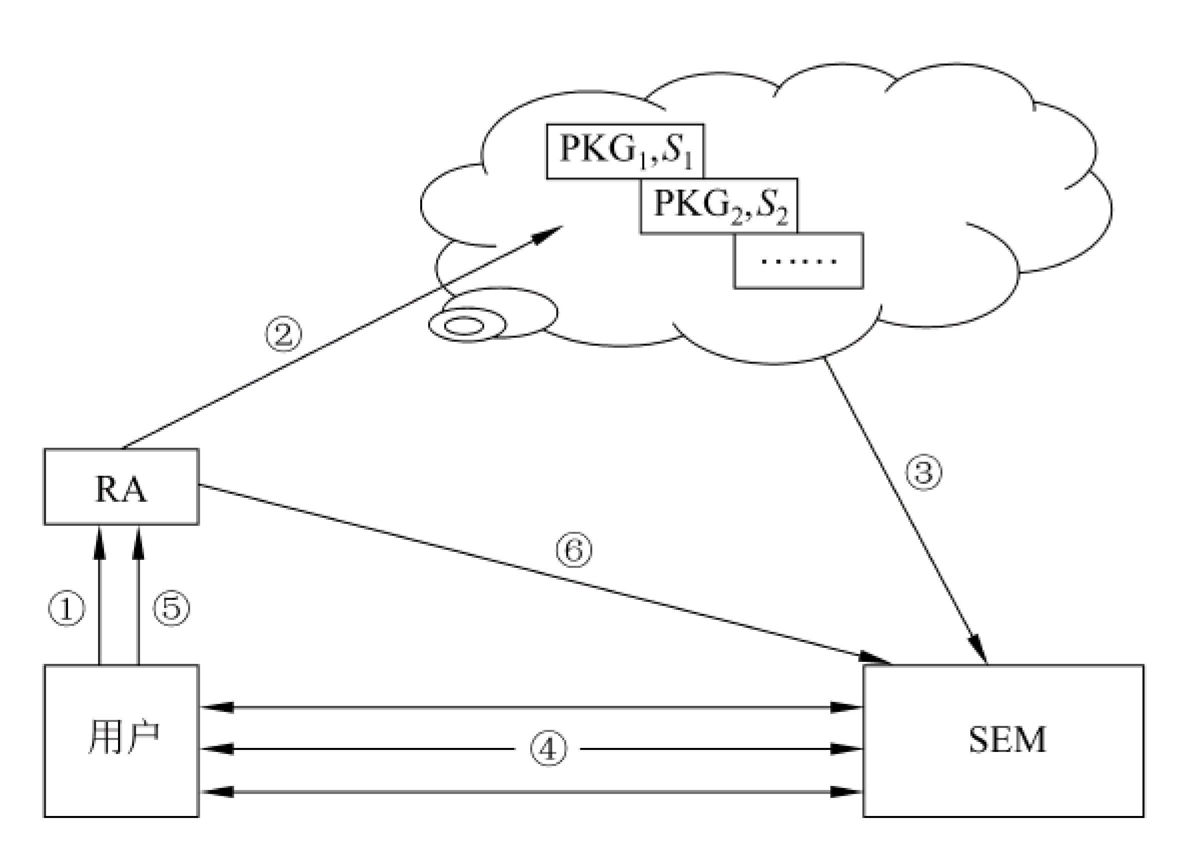


图 15.2　ID-PKI 系统逻辑结构

安全信道。这个假设同传统 PKI 中的假设是类似的。系统中每个 PKG 都掌握一个主密钥 s_i。系统的完整主密钥 s 可由数量不低于门限值 t 的 PKG 使用拉格朗日插值法计算得到，即 $s=\sum\lambda_i s_i$，λ_i 是拉格朗日插值系数，但在任何时候 s 都不被恢复出来。在系统初始化时，各 PKG 使用拉格朗日插值法协同计算出系统公开密钥 $P_{\text{Pub}}=sP$，并公布系统公开参数，系统公开参数与 15.2 节介绍的密钥隔离密码方案中定义的完全相同。需要指出的是，对用户而言，系统中只有一个 PKG，其公开密钥是 $P_{\text{Pub}}=sP$，系统内部多个 PKG 的组织结构对其是透明的。这样，系统内部增加或减少 PKG 的数量或通过前摄的方法[40~43]更新各 PKG 的主密钥对用户不会有任何影响。

2. 密钥分发

密钥分发过程主要包括以下几个步骤。

(1) 用户在申请私钥前，先选择一个随机数 $r\in_R Z_q^*$，并计算 $R=rP\in G_1$ 和 $\text{Sk}_{\text{ID,U}}=rP_{\text{Pub}}$。

(2) 用户以离线方式向 RA 证明自己的身份，并注册身份信息 ID 和随机数 R。

(3) RA 验证用户身份正确后，将 ID 和 R 安全地分发给各 PKG。

(4) 各 PKG 使用自己的主密钥 s_i，计算 $d_i=s_i(Q_{\text{ID}}-R)$，其中 $Q_{\text{ID}}=H_1(\text{ID})\in G_1$，并将结果发送到 SEM。

(5) SEM 接收到不低于 t(t 为门限值)个 d_i 后，计算 $\text{Sk}_{\text{ID,S}}=\sum_1^t\lambda_i d_i$。SEM 将二元组 $(\text{ID},\text{Sk}_{\text{ID,S}})$ 保存在自己的资料库中。

(6) SEM 使用 $\text{Sk}_{\text{ID,S}}$ 定期更新用户私钥，用户使用 $\text{Sk}_{\text{ID,U}}$ 和来自 SEM 的密钥更新信息 $\text{Ku}_{\text{ID},t}$ 计算临时私钥 $\text{Tk}_{\text{ID},t}$。

容易验证，上述过程结束后，用户的完整私钥 d_{ID} 等于两部分 Home key 之和，即 $d_{\text{ID}}=\text{Sk}_{\text{ID,U}}+\text{Sk}_{\text{ID,S}}$，因为有下面的等式成立，即

$$\begin{aligned}\text{Sk}_{\text{ID,U}}+\text{Sk}_{\text{ID,S}}&=rP_{\text{Pub}}+\sum_1^t(\lambda_i d_i)=sR+s(Q_{\text{ID}}-R)\\&=sR+sQ_{\text{ID}}-sR=sQ_{\text{ID}}\end{aligned}$$

在用户申请私钥过程中，即使泄露了 R 的值，也不会影响 $\text{Sk}_{\text{ID,U}}$ 的安全性，因为

$(P, P_{Pub}, R, Sk_{ID,U})$构成了一个 CDH 问题实例。在 CDH 问题难解假设下，只要攻击者不能获得用户选择的随机数 r 或者系统主密钥 s，$Sk_{ID,U}$ 就是安全的。在攻击者不知道 $Sk_{ID,U}$ 的情况下，如果篡改了用户向 RA 注册的 R 值，那么当用户接收到 SEM 发送的第一个密钥更新信息时就发现这种攻击行为(因为密钥更新信息具有“足够”的不可伪造性)，并立即报警，总是假设用户在向 RA 注册后会立即向 SEM 请求密钥更新。

3. 密钥撤销

SEM 在该系统中负责控制所有用户的密钥撤销状态。SEM 只要停止响应用户的密钥更新请求，即可达到撤销该用户密钥的目的。这样在以后的时间段里，该用户不能再解密任何新的密文。

对于用户发起的密钥撤销，其处理流程如图 15.2 中第⑤、⑥步所示。用户需先向 RA 认证身份，身份验证通过后，再经由 RA 向 SEM 请求密钥撤销。这与传统 PKI 中用户的证书撤销流程是类似的。当然用户也可以直接向 SEM 认证身份并请求撤销密钥，但通常情况下，SEM 都是一个远端服务器，而且此时又往往需要离线验证用户身份(如通过身份证、介绍信等)，因此通过本地的 RA 机构请求撤销通常更为实际。该方案的密钥撤销粒度(Revocation Granularity)取决于密钥更新周期的长短，周期越短，密钥撤销粒度就越细。方案中密钥更新的代价非常小，只需要少量运算和一次网络连接，而且不需要认证身份，因此密钥更新周期完全可以设置得非常短，如 1 天或 1 小时。

使用该方案，依赖方可以在加密密文时不必事先查询对方密钥是否被撤销，因为如果对方密钥没有被撤销，则可以正常解密出明文，而如果对方密钥已经被撤销，那么不能获得明文的任何信息。同样地，用户密钥被撤销后，也不能再产生任何新时间段的签名结果。

4. 密钥托管

该方案通过使用多个 PKG 共同产生用户私钥避免了单个 PKG 托管用户密钥的问题，方法与 BF 方案是相同的。该方案使用门限密码技术以防止若干 PKG 损毁导致整个系统崩溃。另外，该系统中虽然设置了多个 PKG，但不会增加用户的任何负担，实际上系统中 PKG 的数量和位置对用户是透明的，用户只需要在初始时向 RA 注册一次身份，之后定期从 SEM 获取密钥更新信息即可，这是该方案同其他多权威机构方案[5,60,63]相比的一个优势。

值得一提的是，密钥托管并非一无是处，在某些情形下反而非常有用，如当公安机关需要解密犯罪分子通信记录时，或者用户需要恢复丢失的私钥时。因此，应预留一种方法使得系统在必要的时候仍能恢复用户私钥。在该方案中，不低于门限数量的 PKG 就可以恢复用户私钥，当然这必须在一定的监督下进行，并通过审计日志的形式保证日后可以追踪。然而，文献[61]和[62]的方案完全避免了密钥托管的可能，即使是“善意”的密钥托管也很难实现。

5. 用户私钥保存

该方案中，用户必须妥善保存 $Sk_{ID,U}$，防止攻击者获取。这里提供两种保存方法。

(1) 在用户申请私钥时先选择一个容易记忆的口令 password，并使用安全的 Hash 函数 H(如 SHA-1 等)计算 $r=H(\text{password})$。这样，用户不需要在设备中存储 Home key $Sk_{ID,U}$，而只需在每次计算临时私钥时，通过输入口令临时计算 $Sk_{ID,U}=H(\text{password})\cdot P_{Pub}$ 并用之计算临时私钥 $Tk_{ID,t}$，之后安全地删除内存中的 $Sk_{ID,U}$。这样，$Sk_{ID,U}$不必在任何设备

中存储，攻击者即使窃取了用户的解密设备，也只能获得当前时间段的临时私钥。

(2) 使用防窜扰的硬件设备生成随机数 r 并保存。当用户申请私钥时，该设备输出 $R=rP$ 并保存 $\mathrm{Sk}_{\mathrm{ID,U}}=rP_{\mathrm{Pub}}$ 在设备中。密钥更新时，该设备通过接口输入密钥更新信息 $\mathrm{Ku}_{\mathrm{ID},t}$，计算后输出临时私钥 $\mathrm{Tk}_{\mathrm{ID},t}$。通过使用口令或生物特征保护，可以保证只有合法用户才能使用该设备。这样，攻击者即使窃取了用户的解密设备，也无法获得或使用 $\mathrm{Sk}_{\mathrm{ID,U}}$。

使用这两种方法，均可保证 Home key $\mathrm{Sk}_{\mathrm{ID,U}}$ 的安全性。当然，在计算临时私钥时，应使用以下计算方法以防止攻击者根据临时私钥和密钥更新信息计算出 $\mathrm{Sk}_{\mathrm{ID,U}}$。

(1) 计算 $H_u=H_u(t)$。

(2) 选择随机数 $v\in_R Z_q^*$，计算 $X'=v\mathrm{Hu}$ 和 $Y'=vP$，并立即删除 v。

(3) 计算 $X_{\mathrm{ID},t}=\mathrm{Sk}_{\mathrm{ID,U}}+X+X'$，$Y_{\mathrm{ID},t}=Y+Y'$，用户临时私钥为 $\mathrm{Tk}_{\mathrm{ID},t}=<X_{\mathrm{ID},t},Y_{\mathrm{ID},t}>$。

使用这种方法，用户在临时私钥中增加了一个随机因子，攻击者在不知道 v 的情况下不能从 $X_{\mathrm{ID},t}$ 分离出 X'，因此也就不能获得用户的 Home key $\mathrm{Sk}_{\mathrm{ID,U}}$。这也是应用 IDKIE 方案的一个安全性技巧。

6. 私钥更换

ID-PKI 中，如果用户因为私钥泄露而请求撤销密钥，则应允许用户在撤销原有私钥后更换一个新的私钥，并且使用新的私钥在以后的时间段正常地解密/签名，同时保持公钥即身份信息不变。

该方案中，可以按照以下步骤更换用户私钥。

(1) 用户重新产生一个随机数 $r'\in_R Z_q^*$（通过更换口令或使用密码设备随机产生），并计算 $\mathrm{Sk}'_{\mathrm{ID,U}}=r'P_{\mathrm{Pub}}$ 和 $\mathrm{RK}=\mathrm{Sk}'_{\mathrm{ID,U}}-\mathrm{Sk}_{\mathrm{ID,U}}$。

(2) 在向 RA 请求撤销时，用户经由 RA 将 RK 转发给 SEM。

(3) 用户更换其 Home key 为 $\mathrm{Sk}'_{\mathrm{ID,U}}$，SEM 更换其 Home key 为 $\mathrm{Sk}'_{\mathrm{ID,S}}=\mathrm{Sk}_{\mathrm{ID,S}}-\mathrm{RK}$。

(4) 在以后的时间里，SEM 使用 $\mathrm{Sk}'_{\mathrm{ID,S}}$ 为用户颁发密钥更新信息，用户使用 $\mathrm{Sk}'_{\mathrm{ID,U}}$ 计算临时私钥。

首先，易知更换后的 Home key $\mathrm{Sk}'_{\mathrm{ID,U}}$ 和 $\mathrm{Sk}'_{\mathrm{ID,S}}$ 仍是有效的，因为等式 $\mathrm{Sk}'_{\mathrm{ID,U}}+\mathrm{Sk}'_{\mathrm{ID,S}}=d_{\mathrm{ID}}$ 仍然成立；其次，只要攻击者不能获得 RK 的值，新的密钥更新信息对其就没有任何意义，即使攻击者掌握了用户旧的部分私钥，也不能用其获得新的临时私钥。

7. 无人看管的密钥更新

该方案的密钥更新过程由于需要使用用户的 Home key $\mathrm{Sk}_{\mathrm{ID,U}}$，因此需要用户的亲自参与（如输入口令或插入密码设备）。但在实际环境中，用户很难时时守候在设备前看管密钥更新。例如，当用户外出或休息时，希望办公室的计算机可以照常接收并解密邮件，这样，就需要计算机能够在无人看管的情况下自动进行密钥更新，获得最近时间段的临时私钥。但同时，又要保证用户 Home key $\mathrm{Sk}_{\mathrm{ID,U}}$ 的安全性。

该方案可以提供这样一个无人看管的密钥更新功能。用户在需要外出时，可以事先选择一个随机数 $r\in Z_q^*$，并计算 $\mathrm{Xtemp}=\mathrm{Sk}_{\mathrm{ID,U}}+rH_u(t)$，$\mathrm{Ytemp}=rP$，其中 t 是下个时间段的字符串表示，用户将(Xtemp，Ytemp)存储在计算机中，当需要密钥更新时，计算机可以按下面的步骤自动检验密钥更新信息的合法性，并生成临时私钥。

(1) 收到密钥更新信息 $\mathrm{Ku}_{\mathrm{ID},t}=<X,Y>$ 后，检查是否有等式 $\hat{e}(\mathrm{Xtemp}+X,P)=$

$\hat{e}(Q_{ID}, P_{Pub}) \cdot \hat{e}(H_u(t), Y+\text{Ytemp})$成立。

(2) 如果该等式成立，则计算临时私钥 $\text{Tk}_{ID,t}=\langle \text{Xtemp}+X, \text{Ytemp}+Y \rangle$。

(3) 否则，密钥更新信息不合法，报告错误。

容易验证，如果密钥更新信息合法，则上述过程可以计算出正确的临时私钥 $\text{Tk}_{ID,t}$。同时基于 r 的随机性和 CDH 问题的难解性，(Xtemp，Ytemp)只能用于检验并产生第 t 时间段的临时私钥，不会危害 $\text{Sk}_{ID,U}$和其他时间段临时私钥的安全。

8. 密码算法和安全性

该方案中用户可以使用 15.2 节给出的 IDKIE 方案和 IDKIS 方案进行加解密和签名运算，系统公开参数也与 15.2 节给出的完全相同。

通过使用上述的用户私钥保存方法可以保证 $\text{Sk}_{ID,U}$ 的安全性。在此基础上，根据文献[33]对 IDKIE 和 IDKIS 方案的安全性分析，可以确信攻击者窃取用户密码设备只能获得当时的临时私钥，而仅截获密钥更新信息不能获得任何时间段的私钥。SEM 虽然掌握用户的部分私钥，但不能获得任何时间段的解密/签名能力。鉴于此，SEM 向用户发送密钥更新信息时，完全可以以明文形式传送。同时，文献[33]中也指出，密钥更新信息自身可以保证“足够”的完整性保护，攻击者即使篡改密钥更新信息，也只能以“无害”的方式篡改，否则用户可以检测出来。

9. 性能分析与比较

令 m 表示 G_1 群上的乘法，在该方案的私钥申请过程中，用户需要计算 $2m$，每个 PKG 需要计算 $1m$，SEM 需计算 tm(t 为门限值)。密钥更新及加密、签名时各方的计算量和通信量在 15.2 节已经分析过，这里不再赘述。

除了提供完整的密钥安全分发、密钥撤销和密钥托管解决方案外，该方案还有以下几个特点。

(1) 用户的部分私钥由用户自己生成，在任何时候都不在网络上传送。

(2) 除非用户密钥被撤销，否则在整个生命周期内，只需要一次认证身份和安全分发过程(用户注册时)。

(3) 用户在获得密钥更新后，可以完全自主地解密、签名，不需要外界帮助。

(4) 如果采用口令方式保存部分私钥，用户部分私钥不需要在计算机设备中存储，由此可以增强系统安全性。

(5) PKG 只与 RA、SEM 交互，不直接面向用户，而且在所有用户都注册后，PKG 可以离线。

(6) 虽然采用多个 PKG 避免密钥托管问题，PKG 的个数和部署对用户是透明的。

从提供的密钥管理功能来看，该方案同现有的其他主要方案的对比参见表 15.1。

表 15.1 功能对比

	BF 方案	LQ 方案	Lee 方案	OLM 方案	Sui 方案	ID-PKI 方案
密钥分发	×	×	√	√	√	√
密钥撤销	√	√	×	√	×	√
密钥托管	√	×	○	○	√	√

“√”表示该方案具有此功能；“×”表示该方案不具有此功能；“○”表示该方案虽提供此功能，但存在不足。

下面从性能方面同现有的几个主要方案进行一一对比分析。

首先同 BF 方案相比，ID-PKI 方案不必为了提高密钥撤销粒度，缩短用户私钥有效期，而且除非需要撤销密钥，否则在整个密钥生命周期内用户只需要认证一次身份，且不需要保密发送密钥更新信息。而 BF 方案中，为了提高密钥撤销粒度，密钥有效期必须比较短，而且每次颁发私钥都需要认证用户身份，并秘密发送私钥。另外，由于需要频繁颁发私钥，BF 方案中 PKG 必须总是在线，而 ID-PKI 方案中，PKG 在完成用户初始注册后即可离线，而且 PKG 不需要直接面对用户，因此 ID-PKI 方案中的 PKG 更安全。但 ID-PKI 方案运算量和通信量比 BF 方案略增。ID-PKI 方案中的用户加密比 BF 方案中多一次 G_1 上的乘法，解密比 BF 中多一次 $\hat{e}$ 运算，密文长度多$|G_1|$，$|G_1|$表示 G_1 群上一个点的二进制表示长度。

LQ 方案可以实现即时撤销，该方案则难以实现这一点，但该方案可以通过缩短密钥更新周期提高密钥撤销粒度(如 1 小时)，完全能够满足大多数应用环境的需要。ID-PKI 方案中用户一个时间段只需要同 SEM 通信一次，然后在整个时间段内都可以离线解密/签名；而 LQ 方案中，用户每次解密/签名都需要同 SEM 交互，因此，不能离线进行任何解密/签名。该方案中用户加密过程比 LQ 方案中多一次 G_1 上的乘法，解密过程的计算量相当，但因为不需要同 SEM 交互，因此速度更快，且节省通信量。LQ 方案在进行概率签名时需要 SEM 和用户分布产生随机数，难度较大，而 ID-PKI 方案中用户自己产生签名，不必考虑分布产生随机数的问题。另外，LQ 方案中，用户在解密出错时，很难判断是密文的错误还是 SEM 中间结果的错误，而 ID-PKI 方案中用户可以检查 SEM 密钥更新信息的合法性。

OLM 方案继承了 Lee 方案中的密钥托管解决办法，下面将其一起比较。首先 OLM 方案和 Lee 方案都没有完全解决密钥托管问题，因为 PKG 完全可以假冒用户向 KPA 获取用户的其他部分私钥。其次，这两个方案中，系统的主密钥相当于 PKG 和所有 KPA 的部分主密钥的乘积，需要 PKG 和所有 KPA 串行计算用户私钥，耗时较多。另外，由于没有冗余，任何一个部件损坏都会导致系统崩溃。OLM 方案的撤销机制与 LQ 方案中相同，与该方案的性能比较不再赘述。

Sui 方案的密钥分发方案与 ID-PKI 方案较为类似，但 Sui 方案中用户必须知道所有 KGC 的数量和位置，而且需向所有 KGC 都索取私钥信息，而 ID-PKI 方案中 PKG 对用户而言是透明的，并且用户不需要同 PKG 联系。另外，Sui 方案中没有考虑密钥撤销问题，如果使用 BF 撤销方案的话，用户每个时间段都需要调用一次 Sui 方案向 RA 注册口令并从多个 KGC 秘密传送私钥，因此效率太低；而 ID-PKI 方案用户只需要初始注册时向 RA 验证身份并注册参数，以后的时间段从 SEM 接收密钥更新信息即可。与 ID-PKI 方案相比，Sui 方案的优势在于可以使用现有的许多 ID-PKC 方案如 Boneh-Franklin IBE 等，而 ID-PKI 方案则不能直接使用。

15.3.3 可抵抗主动攻击者的 ID-PKI 密钥管理方案

15.3.2 小节介绍的方案的安全性基于攻击者不能获得用户部分私钥 $Sk_{ID,U}$。在此前提下，密钥更新信息可以在网络中以明文形式传送。但攻击者一旦获得了 $Sk_{ID,U}$，就可以通过窃听密钥更新信息获得同合法用户一样的解密/签名能力。而且用户很难在短时间内发现

这种攻击行为。

为了避免攻击者获得 $Sk_{ID,U}$，在 15.3.2 小节提供了两种用户部分私钥保存方法，一种使用口令方式保存，另一种使用防窜扰密码设备保存。但在一些应用环境中，这两种方法是不够的，或者是不适用的。如用户可能使用了弱口令，被攻击者猜测到，或者用户终端没有条件使用防窜扰设备。因此，在有些场合下不能排除攻击者获得用户部分私钥的可能。在这种情况下仅使用 TLS/SSL 或者其他密钥协商协议保护密钥更新信息是不够的，因为主动攻击者侵入用户设备并获取了用户所有私钥后，可以假冒用户向 SEM 请求密钥更新。15.2.3 小节给出了一个抵抗此类主动攻击者的 AR-IDKIE 方案。本小节基于 AR-IDKIE 方案给出了一种可以抵抗主动攻击者的 ID-PKI 密钥管理方案。

该方案允许攻击者侵入用户设备并获得用户所有私钥信息，并允许其假冒用户向 SEM 索取密钥更新信息(但假设其在一定时间内只能索取有限次密钥更新信息)。在这种情况下，掌握了用户当前所有私钥信息的攻击者，如果假冒用户向 SEM 请求密钥更新，则会被用户在之后请求密钥更新时发现；如果被动监听密钥更新消息，则无法获得任何时间段的临时私钥，而且其掌握的部分私钥也不再有任何意义。

1. 系统结构

ID-PKI 密钥管理方案的系统结构与 15.3.2 小节是相同的，都包含 RA、SEM 和多个 PKG，如图 15.2 所示。假设用户的部分私钥存放在终端设备的物理存储介质上(如硬盘)，攻击者可能通过短暂侵入用户设备获得其中存放的所有用户私钥信息。攻击者还能够控制用户同外界的所有网络连接，获得其中传输的所有消息，并能够篡改、扣留、发送消息。但假设攻击者不能控制用户同 RA 的离线认证与通信，而且，用户在发现私钥被攻击者窃取后，会立即报警，并采取补救措施挽回损失。

该方案的密钥分发过程与 15.3.2 小节中的基本相同，但在用户向 RA 认证身份并注册身份信息和 R 后，PKG 为用户额外生成一个用于在密钥更新阶段认证身份的认证私钥 AK_{ID}，并通过 RA 转发给用户。为了避免认证私钥被用作其他用途，可以在其公钥信息上附加密钥用法，如“Alice ‖ For key update authentication only”。从 RA 向用户发送认证私钥的过程是在用户离线向 RA 注册时进行的，总假设这个离线过程是安全的。SEM 同样有一个用于密钥更新的认证私钥，记为 AK_{SEM}。使用 AK_{ID} 和 AK_{SEM}，用户与 SEM 可以安全地进行前摄密钥更新，更新过程在前面已给出。用户按照前面的 AR-IDKIE 方案加、解密消息。该方案的密钥撤销和密钥托管解决方案同 15.3.2 小节中完全相同，这里不再重复。

2. 安全性分析

该方案的安全性除了基于 AR-IDKIE 方案安全性之外，还基于 PKG、RA 和 SEM 3 个系统部件自身的安全性。AR-IDKIE 方案的安全性已在文献[33]中讨论过，这里主要讨论该方案对 3 个系统部件安全性的依赖。

首先，PKG 作为整体而言必须绝对可信，然而就局部而言，少于门限值的 PKG 背叛并不能损害系统的安全性。采用门限密码技术降低了系统对单个 PKG 安全性的依赖。

其次，RA 在整个系统中负责验证用户身份，而且可以获得所有用户的认证私钥，因此 RA 必须是可信的。然而，RA 即使是恶意的，也不能获得用户生成的部分私钥信息；虽然 RA 可以主动替换用户注册的 R 值以获得用户部分私钥，但用户在第一次请求密钥更新时

就会发现 RA 的这种攻击行为。

最后，SEM 作为控制整个系统用户密钥撤销状态的系统部件，必须是安全可信的，这与所有密钥撤销方案都是类似的。另外，对 AR-IDKIE 方案而言，攻击者如果获得了 SEM 的认证私钥以及其中存储的用户部分私钥，就可以向用户假冒 SEM，然后再伺机侵入用户设备获得用户另一部分私钥，而且在这之间用户和 SEM 都很难发现该攻击者存在。但攻击者仅获得 SEM 中的部分私钥，而不会获得任何时间段的解密/签名能力。

3. 性能分析

该方案中加、解密效率和密文长度同 15.3.2 小节的方案是完全相同的，密钥更新的效率取决于所使用的 AKC 方案的效率，如果使用文献[25]中的 ID-AKC 协议，则密钥更新效率分析参见前面章节。

该方案在具备 15.3.2 小节密钥管理方案的所有功能的同时，还可以使用户能够及时发现部分私钥被窃取，或者摆脱攻击者对部分私钥的控制。需要指出的是，在攻击者能够获得用户所有私钥信息，并能主动发起攻击的情况下，是不可能阻止攻击者假冒用户获取密钥更新信息的，因此，对此种攻击者最优的防护效果就是能够在短时间内发现其存在，并采取措施清除攻击者。该方案中用户可以在一个密钥更新周期内发现主动攻击者，或者清除其对用户部分私钥的控制。

同现有其他密钥管理方案相比，该方案除了具有 15.3.2 小节密钥管理方案已有的优势外，还能够抵抗主动攻击者的攻击。而现有的密钥管理方案都没有考虑主动攻击者的问题。比如在 LQ 方案中，攻击者如果可以获得用户方的部分私钥，就可以假冒用户向 SEM 请求帮助解密或签名，在合法用户没有发现该攻击者存在并撤销密钥之前，SEM 总会响应攻击者的要求。然而，通常情况下，只有造成重大损失后，用户才会发现自己私钥被窃取。在 BF 方案中，用户和 PKG 必须有一个安全信道传送用户私钥。如果采用在线传送方式，攻击者在获得用户认证凭证（如口令、私钥）后，就可以假冒该用户获取以后所有时间段的私钥，而且不容易被发现。

15.4 基于身份的多信任域网格认证模型

除了密钥管理问题以外，ID-PKI 系统研究的另一个重要问题是如何将其应用到实际环境中，并充分发挥 ID-PKI 的优势。本节分析了现有网格认证框架存在的问题，在 15.3 节构造的 ID-PKI 密钥管理方案的基础上，给出了一个基于身份的多信任域网格认证模型。考虑到网格多信任域的特点，该模型提供了跨信任域的双向实体认证。与基于传统 PKI 的网格认证框架相比，该方案更轻量、更高效；与现有的基于身份网格安全框架相比，该方案更好地解决了密钥撤销、分发和托管等问题，因而更具实用性。

15.4.1 研究背景和相关研究进展

网格技术通过互联网把分散的计算资源整合成一个虚拟组织，进而实现资源共享和协同工作。虚拟组织由多个异构的信任域组成，每个域都是独立、自治的。由于网格环境的复杂性以及网格所涉及的巨大利益，安全成为网格体系的一个重要方面。其中，作为网格安全的基础，认证显得尤为重要。

在网格中,由于计算资源和存储资源的分散性,用户的每次请求都可能需要由分散在若干信任域中的多个服务器协同完成。同时,网格中的资源请求都是在运行过程中动态进行的,具有不可预测性。这些特性决定了网格认证应满足以下要求[69,70]。

(1) 支持不同信任域间的双向实体认证。

(2) 支持单点登录(Single sign-on)。

(3) 轻量、高效。

现有的许多网格安全体系如 GSI、OGSA、Akenti[70~72]等,都以传统 PKI 为基础构建认证框架,通过 SSL 认证协议(SSL Authentication Protocol,SAP)实现双向身份认证。这种认证框架比基于对称密码的认证框架(如 Kerberos)有了较大的进步[69],但仍存在 SAP 认证效率过低、PKI 系统建设和维护成本太高等不足。

造成 SAP 效率低的原因主要有两个:一是认证过程中的证书处理工作过于繁重,尤其是证书状态查询和证书路径构造,造成较大的计算负担和时间延迟[6,73];二是认证过程中双方需要交换证书链,造成较大的通信开销。对于资源申请非常频繁的网格而言,上述两点都使得 SAP 容易成为系统的瓶颈。PKI 系统成本高则是目前 PKI 应用普遍遇到的问题。基于证书的 PKI 系统由于需要提供证书查询、证书申请、证书撤销等诸多服务,造成系统结构过于复杂,建设和维护成本过高。显然,这种认证框架不能满足"轻量、高效"的要求。

然而,使用 ID-PKI 却可以比较容易地避免上述问题。由于不再需要证书,基于 ID-PKI 的应用可以避免证书处理和证书传递造成的开销,同时也使 PKI 系统结构大大简化。ID-PKI 的这些优点,使得在此基础上构建的认证框架可以比现有的认证框架更轻量、更高效,因而能更大程度地满足网格认证的需要。

近年来随着 ID-PKC 的迅速发展,利用 ID-PKC 的优势解决网格安全问题的思路逐渐引起关注,出现了若干解决方案[74~78],后面将对这些方案简要介绍。但正如前面所分析的那样,ID-PKC 虽然有较大优势,但为了在此基础上构建真正实用的 ID-PKI 系统和应用,必须妥善解决密钥分发、密钥撤销和密钥托管等问题。然而,现有基于 ID-PKC 的各种网格安全方案[74~78]大都侧重于考虑如何应用 ID-PKC 解决具体应用问题,对上述的密钥管理问题却考虑不够。

文献[78]中给出了一种基于身份的非交互网格认证框架。该框架中,认证双方使用文献[10]中的非交互密钥分发方案建立一个共享的会话密钥。使用这个会话密钥认证双方可以实现相互认证,并保护会话的机密性和完整性。该方案假设网格中所有实体的私钥都由同一个 PKG 生成。然而通常情况下,网格系统包含多个独立的自治域,要求所有自治域都信任同一个 PKG 显然不现实。该方案中,认证双方建立的会话密钥是一个静态密钥,如果攻击者获得该密钥,就可以完全掌握双方的会话内容,并可以假冒其中的任一方。因此,文献[78]中的认证框架仍难以完全满足网格认证的要求。

文献[76]中讨论了 ID-PKC 在网格应用中的优势,并建议使用 ID-PKC/PKI 混杂模式构建网格安全基础设施,即每个域都有一个 PKG 为用户颁发 ID-PKC 私钥,在全体域之上存在一个传统 PKI 的 CA,为各域的 PKG 颁发证书。同一个域内的用户可以直接使用 ID-PKC 方案进行认证和安全通信,不同域的用户则需使用传统 PKI 的方法先验证对方 PKG 的证书,再实现认证。文献[77]中又对这种 ID-PKC/PKI 混杂模式做了改进,在这个改进方案中每个用户都相当于 ID-PKC 中的一个 PKG,CA 为每个用户颁发证书,证书中包

含该用户的公开参数，用户通过颁发临时私钥的方式实现权限委托(Right Delegation)和单点登录。文献[76]和[77]给出的这两个方案都需要传统 PKI 的支持，虽然在一定程度上体现了 ID-PKC 的优势，但仍不能完全摆脱传统 PKI 的弊端。文献[74]中也讨论了使用 ID-PKC 改善GSI[70]的性能和可扩展性。

文献[75]中使用 HIBC(Hierarchical Identity-Based Cryptography)方案构造了一个网格安全框架(以下简称 LP 方案)，在这个框架中，PKG 使用 HIBC 密钥生成方法为用户生成一层私钥作为长期私钥，用户使用该私钥自行生成两层临时私钥，并使用这个私钥实现权限委托和单点登录。该方案使用 HIBC 改造 SSL 握手协议实现双向实体认证。该方案完全建立在 ID-PKC 基础之上，不再需要传统 PKI 支持。但该方案在使用 HIBC 的同时，也不可避免地继承了 HIBC 的问题，如实现密钥撤销困难，而且如果采用 BF 方案中的密钥撤销方法，就又带来了频繁分发用户私钥的安全和效率问题。

15.4.2 基于身份的密钥隔离 SAP 协议

SAP 专指 TLS/SSL 协议中用于协商密钥的握手协议。TLS/SSL 协议被设计为传输层的子层，为应用层协议(如 HTTP 等)提供安全连接服务。在 GSI[70]等网格安全体系中广泛使用 TLS/SSL 协议实现双向认证，但这种认证并不仅局限于传输层，因此文献[70]将其用于认证的握手协议单独抽取出来，称之为 SAP(SSL Authentication Protocol)协议。

SAP 协议基于传统 PKI，双方通过交换公钥证书并使用公钥加密/签名算法(如 RSA 算法)认证协商一个会话密钥，并实现双向的实体认证。SAP 允许认证双方协商选择密钥协商方式、加密算法及 Hash 函数等，因而具有良好的适应能力，得到了广泛应用。但正如前面分析的那样，为了实现双向认证，SAP 需要认证双方交换各自的证书链、构造证书路径并验证所有证书状态，给认证双方带来较大的计算负担和通信负担。正因为如此，在实际实现时，开发者或使用者往往干脆省略证书状态验证等耗时的操作，但显然，这也造成了许多安全隐患。

本节利用 15.2 节的 IDKIE 方案和 IDKIS 方案对传统 SAP 协议进行改进，构造了 IDKISAP 协议，使其摆脱对证书的依赖。基本思想是：用 IDKIE 和 IDKIS 方案替代 SAP 协议中基于证书的加密和签名算法，从而不再需要与证书相关的传递、验证等操作，协议其他部分与 SAP 基本相同。具体的消息流程如图 15.3 所示。

(1) ClientHello 消息：该消息与 13.4.4 小节中的定义是相同的，其中包含了协议版本号、客户端支持的密码算法 cipher suites、会话 ID 以及客户端产生的随机数 Client random。

(2) ServerHello 消息：收到 ClientHello 消息后，服务器产生与 ClientHello 类似的 ServerHello 消息，包含协议版本号、服务器选择的密码算法、会话 ID 以及服务器产生的随机数 Server random。

(3) ServerIdentifier 消息：该消息是可选的，用于向客户端传送服务器的身份信息。按照 15.2 节的 IDKIE 和 IDKIS 方案中的描述，这里的身份信息可表示为“ServerName ‖ ValidityPeriod”(“‖”表示字符串的合并)，即除了服务器身份标识外，还包括一个密钥有效期。不过基于 ID-PKC 的优势，客户端可以根据应用环境中的特定规则猜测服务器的身份信息，这样，只要应用环境中有明确的身份信息定义规则，ServerIdentifier 消息就不必传送。比如网格环境中可以规定使用“计算机名@域名”的规则定义 ServerName，并规定服务器密钥有效

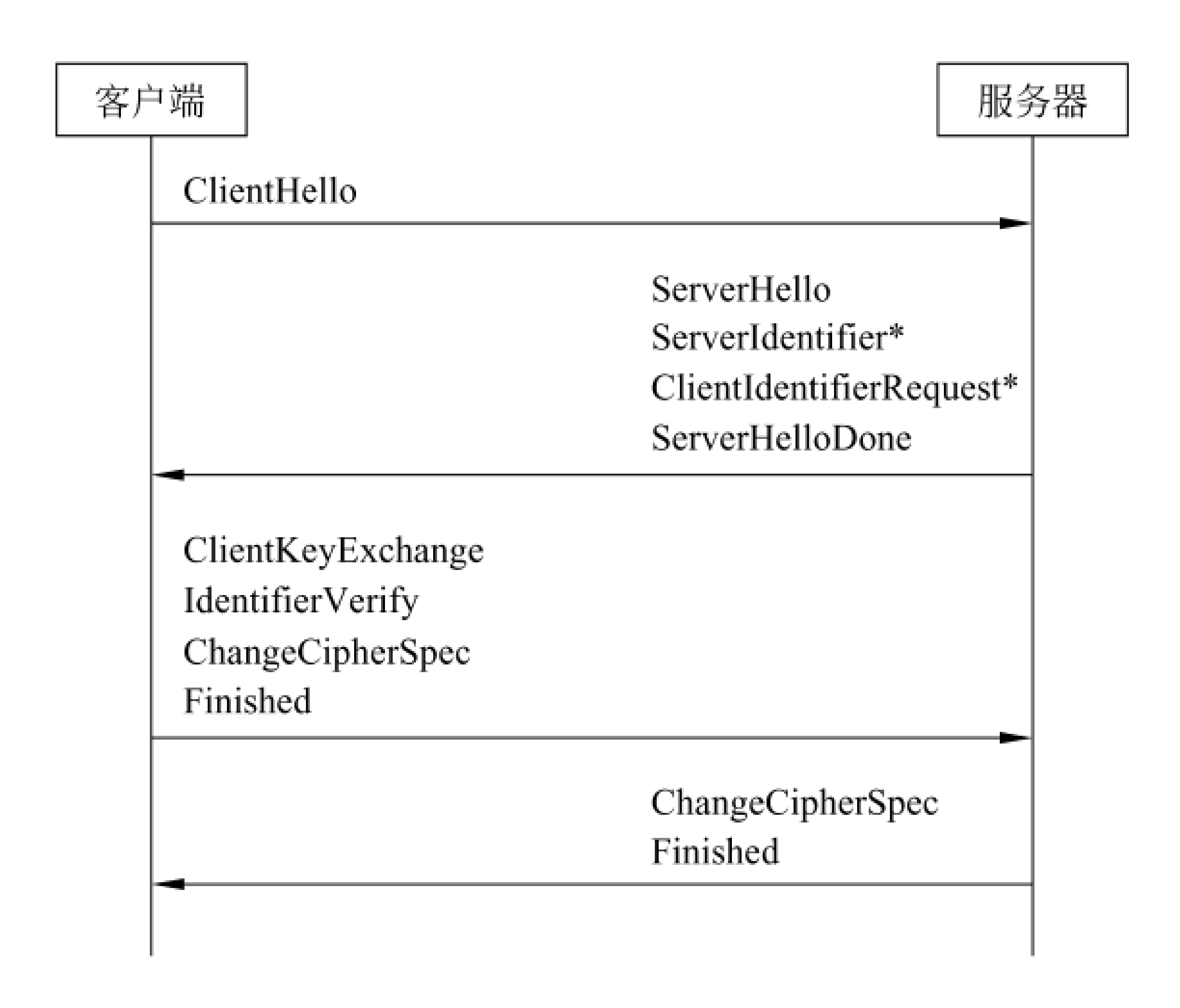


图 15.3 IDKISAP 消息流程示意图

*——可选的消息或根据具体情况决定是否要发送的消息。

期为 1 年，则一个符合该规则的 ServerIdentifier 例子为“WebServer@is. iscas. ac. cn ‖ 2009”。

(4) ClientIdentifierRequest 消息：该消息也是可选的，服务器使用该消息描述客户端身份信息的表达规则。只有应用环境没有定义通用的客户端身份信息表达规则，或者服务器有特殊要求时才使用该消息。如服务器要求客户端的公钥信息除了包含身份和有效期外，还要求客户端在其本域的角色为管理员，则该消息可以表示为“clientIdentifier ‖ role=Administrator ‖ 2009-12-31”。收到该消息后，客户端按照该规则向 SEM 请求密钥更新信息。使用这种方式，服务器可以在认证的同时实现一定的访问控制功能。

(5) ServerHelloDone 消息：该消息与 13.4.4 小节中的定义是相同的，标识 ServerHello 消息结束。

(6) ClientKeyExchange 消息：客户端生成 Premaster secret，并使用 ServerIdentifier 消息中的服务器身份信息，或根据规则“猜测”出的服务器的身份信息加密，加密算法使用 15.2 节的 IDKIE 方案。

(7) IdentifierVerify 消息：客户端使用自己的私钥对之前的交互消息签名，签名算法使用 15.2 节的 IDKIS 方案。签名私钥对应的客户端身份信息应符合 ClientIdentifierRequest 消息或系统通用规则的要求。

(8) ChangeCipherSpec 消息：该消息与 13.4.4 小节中的定义是相同的，用于通知对方开始使用协商的密码算法和密钥。

(9) Finished 消息：该消息与 13.4.4 小节中的定义是相同的，用于向对方确认密钥协商和认证成功完成。Master secret 和会话密钥按照 SSL 协议中的方法生成。

15.4.3 基于身份的多信任域网格认证模型

1. *系统结构*

在该模型中，网格包含多个独立自治的信任域，网格中的用户和资源分散在这些信任域

中。每个信任域都有自己的 PKG 为本域的实体颁发私钥。按照 15.3 节中的 ID-PKI 密钥管理方案，每个信任域除了 PKG 外，还包含 RA、SEM 等配套系统部件。用户和资源服务器间的相互认证是网格计算中最典型的认证活动，为此，这里以跨信任域间的用户-服务器认证为例研究跨信任域的实体认证，其他情况与之类似。该模型的系统结构如图 15.4 所示。

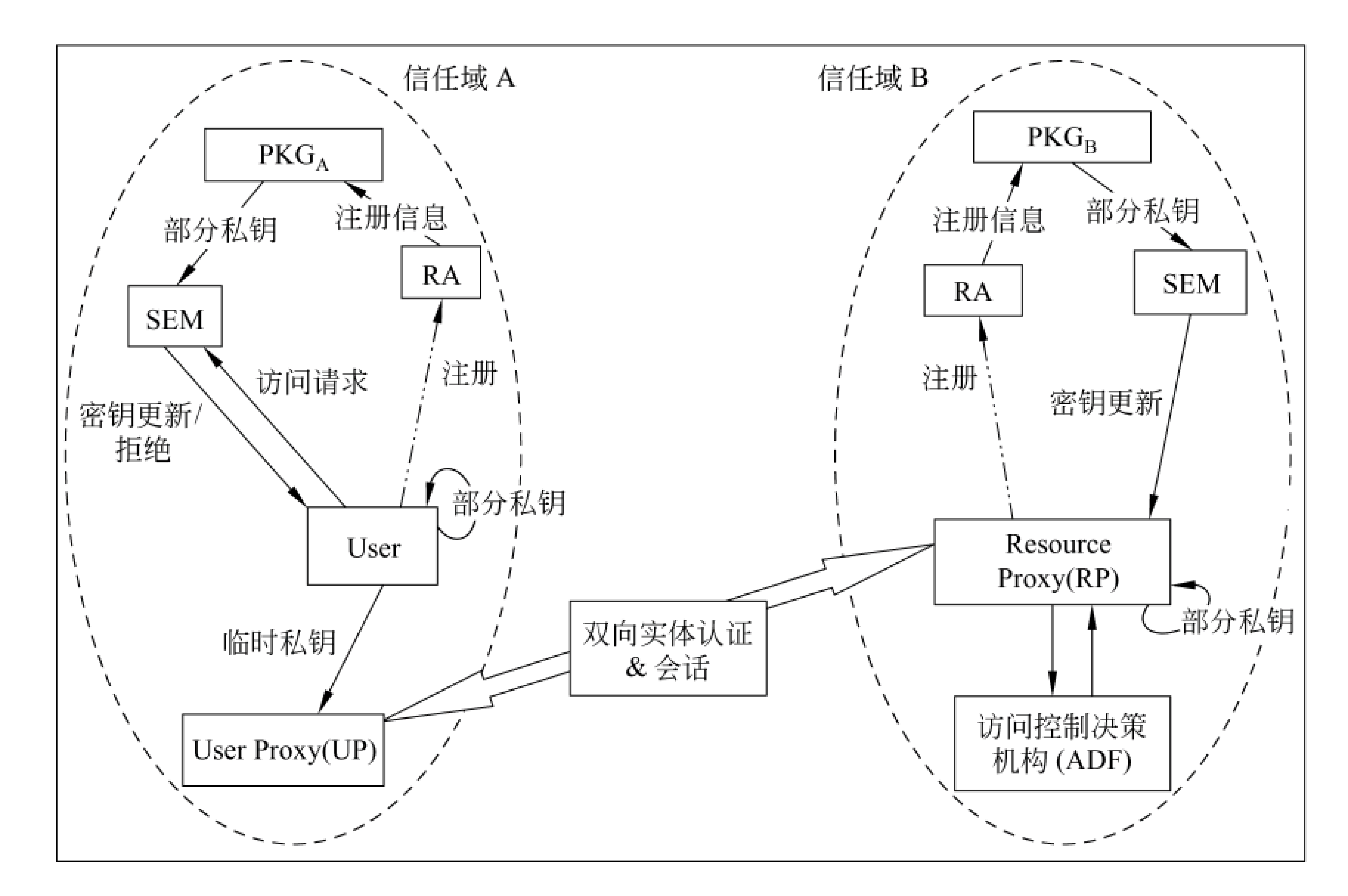


图 15.4　基于身份的多信任域网格认证模型

为了实现单点登录，引入文献[70]中用户代理和资源代理的概念。

(1) 用户代理(User Proxy，UP)，是用户端的一个会话管理进程，可以在一段时间内代表用户进行操作。

(2) 资源代理(Resource Proxy，RP)，是处于资源服务器端的一个代理部件，负责在域间的安全操作和域内的安全机制之间提供转换。

用户代理作为客户端进程在用户计算机上运行，代表用户向资源服务器认证身份；资源代理作为一个服务器端进程，或作为一个系统部件，管理本地资源，处理用户访问请求。资源代理除了认证功能外，往往还承担部分访问控制功能，与访问控制决策机构(ADF)交互，执行访问控制策略。

用户在加入网格时，按照 15.3 节密钥管理方案中的步骤生成部分私钥并向 RA 注册私钥信息，RA 将用户身份和用户注册的私钥信息交给 PKG，各 PKG 使用自己的主密钥分片分别计算中间结果并交给 SEM，SEM 最终计算出用户另一部分私钥。此后，当用户使用网格服务时，先向 SEM 请求密钥更新，然后使用自己的部分私钥计算出下一个时间段的临时私钥，并将临时私钥交给 UP，由 UP 代表用户向各 RP 认证身份并获取资源。

2. 双向实体认证

该模型中，会话双方使用前面给出的 IDKISAP 协议实现双向认证，并认证协商一个会

话密钥，之后，双方使用会话密钥保证通信的完整性和机密性。

该方案的一个前提假设是，网格中所有用户、服务器都掌握各信任域 PKG 的公开参数，并且以不被篡改的形式保存。对于一个网格系统而言，信任域的数量是有限的，而每个信任域的公开参数只占很小的存储空间（约 1KB），因此，就目前 PC 和服务器的存储能力而言，这个假设是可行的。

可以进一步说明其可行性的是，目前的网格安全体系大都基于 PKI 实现。在这种安全体系中，每个结点（用户或服务器）都需要存储大量的用户证书和服务器证书，而每个采用 ASN.1 DER 格式编码的 X.509 证书（使用模长为 1024b 的 RSA 密钥）通常需要约 1KB 的存储空间。因此，相比之下，基于 ID-PKI 的网格认证模型可以更节省用户的存储空间。

3. 单点登录

单点登录是指用户使用网格时，只需要在初始时认证一次自己的身份，在此后的网格计算过程中都不再需要亲自参与身份认证[70]。单点登录可以把用户从网格计算中解放出来，使得网格运行不需要用户一直看管，因此对网格意义重大。实现单点登录的关键在于授予用户代理一个临时性的用户身份凭证，使其能代表用户进行后续的认证；同时又能限制用户代理的能力，使之不会被滥用。

使用 ID-PKI 系统和 IDKISAP 协议，可以很容易地实现用户单点登录。假设用户按照 15.3 节介绍的口令方式或者防窜扰密码设备方式保存部分私钥 $Sk_{ID,U}$。当需要使用网格服务时，用户可以先向 SEM 索取下一个时间段密钥更新信息，然后通过输入口令或者插入密码设备（如 IC 卡）使用部分私钥 $Sk_{ID,U}$ 计算出临时私钥 $Tk_{ID,t}$，并将 $Tk_{ID,t}$ 交给用户代理 UP。之后，UP 可以使用 $Tk_{ID,t}$ 在接下来的时间段内代表用户向各个资源代理认证身份，而不需要用户参与。但 UP 的这种能力仅限于一个时间段内，该时间段过后，除非用户再授予其新的临时私钥，否则 UP 不能再代表用户进行任何认证。这样，既实现了不需用户看管的实体认证，又限制了 UP 的能力，降低了密钥失窃的风险。另外，使用 15.3 节的无人看管的密钥更新方法，用户可以一次性地授予 UP 多个时间段的代理权限，从而能够动态地控制 UP 的能力。

值得注意的是，在响应密钥更新请求时，SEM 不需要验证用户身份，因为在不掌握用户部分私钥 $Sk_{ID,U}$ 的情况下，密钥更新信息对攻击者毫无意义，而基于 RA 的安全性和 15.3 节用户私钥保存方式的安全性，只有合法用户才能掌握该部分私钥。这样，请求密钥更新的工作完全可以由 UP 自动完成，而用户只需要在开始使用网格服务时运行 UP 程序并输入一次口令或插入密码设备，此后就可以完全不再看管，直到网格服务完成。

在资源服务器端，情况是类似的。但由于资源代理需要不间断地提供服务，而且通常情况下服务器端的安全保护强度较高，所以资源代理的私钥的有效期可以较长，以避免频繁更新私钥带来的额外负担。

4. 密钥更新与角色管理

该模型除了可以实现网格认证服务外，还可以在一定程度上支持授权和访问控制的实现。由于网格用户和资源数量众多，使用基于角色的访问控制（RBAC）[79] 实现网格授权管理可以比直接使用用户-权限关联的访问控制模型大大降低授权管理的复杂性，而且更能满足网格动态性的要求[80,81]。

但以何种形式表达用户的角色属性是网格安全研究需要考虑的一个重要问题。传统网格安全框架通常使用属性证书表达用户的角色信息[72]，并通过属性权威(AA)签名保证其真实性。然而在基于 ID-PKI 的网格安全模型中显然不适宜使用属性证书，而且由于使用属性证书需要资源代理验证属性证书的合法性，从而也导致了资源代理工作量的增加。另外，在实际应用中，负责身份管理的系统部件(如传统 PKI 中的 CA)同负责属性管理的系统部件(如 AA)往往是分离的，而且用户身份的变化与其角色的变化通常不同步，因此，在申请私钥时直接在公钥信息中嵌入用户角色是不合适的，因为一旦用户角色变化，用户就需要重新申请私钥。

该模型中，使用密钥更新信息可以容易地表达用户的角色信息，而且不增加资源代理的工作量。为了实现这一功能，SEM 除了管理用户的密钥撤销状态，还通过维护一个用户-角色映射表来管理用户的角色信息。用户向 SEM 请求密钥更新信息时，可以在请求信息中包含角色 $Role_{req}$，SEM 根据用户身份信息 ID 和角色 $Role_{req}$ 检索用户-角色映射表，如果该映射表中存在(ID，$Role_{req}$)这一项，则 SEM 使用该用户的部分信息 $Sk_{ID,S}$ 计算密钥更新信息 $Ku_{ID,t}=(Sk_{ID,S}+xH_u(t,Role_{req}),xP)$，其中 $x\in_R Z_q^*$。用户使用这个密钥更新信息计算出临时私钥 $Tk_{ID,t}$，并交给 UP 用于在下一个时间段代表自己认证身份，此时用户的公钥包含三部分内容：用户身份、有效期和用户的角色信息。UP 使用 IDKISAP 协议同 RP 完成认证后，RP 就可以确信该用户的确具有所宣称的身份和角色。这样，使用 IDKISAP 协议，RP 在认证了用户身份的同时也确认了用户的角色属性，而且认证双方都不增加任何计算量和通信量。

5. 密钥撤销

由于用户仅掌握部分私钥，没有 SEM 的帮助，用户不能产生任何新时间段的临时私钥，也就不能再使用网格服务。如果密钥撤销是由用户私钥泄露造成，则用户可以使用 15.3 节的方法更换部分私钥，之后，用户可以继续使用网格服务，而攻击者即使掌握了用户旧的部分私钥，也不能假冒用户身份使用网格服务。

15.4.4 性能分析与比较

本小节主要分析利用 15.4.3 小节提出的模型进行跨域双向认证时的效率问题，并将之与 SAP 比较。对于 SAP 而言，其认证效率受证书链长度的影响。显然，证书链越短，认证的效率越高。对于不同信任域的双向认证，证书链最短的情况即双方证书链各包含 3 个证书：CA_1 证书、用户证书、用户代理证书以及 CA_2 证书、资源服务器证书、资源代理证书，其中 CA_1 和 CA_2 交叉认证。下面就以这种情况下的 SAP 与 IDKISAP 进行比较。

首先考虑 IDKISAP 协议的通信量，其中 ClientHello 消息和 ServerHello 消息的结构同 SAP 中的定义完全相同，每个消息约 75B。ServerIdentifier 消息和 ClientIdentifierRequest 消息为可选消息，用于表达 Server 和 Client 的身份信息，每个消息约为 100B。ServerHelloDone 消息所消耗通信量可以忽略。ClientKeyExchange 消息和 ClientIdentifierVerify 消息的长度分别由 IDKIE 密文长度和 IDKIS 签名长度决定，而 ClientKeyExchange 消息长度约为 140B，ClientIdentifierVerify 消息长度约为 60B。另外，ChangeCipherSpec 消息和 Finished 消息长度分别为 1B 和 12B。总之，使用 IDKISAP 协议完成一次双向认证共需消耗通信量约

580B，当不使用 ServerIdentifier 消息和 ClientIdentifierRequest 消息时，通信量降为 380B。

接下来分析 SAP 协议的通信量，首先假设 CA 证书和用户证书都使用模长 1024b 的 RSA 公钥，证书以 ASN.1 DER 格式编码，则每个证书大小约 1KB。为了实现双向认证，Client 和 Server 需要相互传递各自的证书链，每个证书链包含 3 个证书，这样 ServerCertificate 消息和 ClientCertificate 消息各需约 3KB。ClientKeyExchange 消息使用 1024b RSA 公钥加密 Premaster secret，Client CertificateVerify 消息使用 1024b 为 RSA 签名，各需至少 128B。这样，使用 SAP 协议完成一次双向认证共需约 6500B。

在 IDKISAP 协议中，主要计算量包括客户端的 IDKIE 加密运算和 IDKIS 签名运算，以及服务器端的 IDKIE 解密运算和 IDKIS 签名验证运算。用 p 表示 pairing 运算，m 表示 G_1 上的乘法，e 表示 G_2 上的幂运算，则客户端的运算量为 $4m+1p+1e$，服务器端运算量为 $6p+1e$。

SAP 协议中，Client 端的主要计算量包括 Client 端对 Server 证书链的验证、对 Premaster secret 的 RSA 加密运算和一次 RSA 签名运算，其中对证书链的验证包括至少 3 次签名验证运算，这样 Client 端主要计算量包括 4 次 RSA 加密运算(明文消息长度小于模长时，RSA 加密速度与签名验证运算速度基本相当)和一次 RSA 签名运算。Server 端的主要计算量包括对 Client 证书链的验证、对 Premaster secret 的解密运算(明文消息长度小于模长时，RSA 解密速度与签名运算速度相当)和一次 RSA 签名验证运算，共计 4 次 RSA 加密运算和一次 RSA 签名运算。

IDKISAP 协议和 SAP 协议效率的对比参见表 15.2。

表 15.2　IDKISAP 与 SAP 效率对比

通信量(B)			计　算　量
IDKISAP	客户端	380	$4m+1p+1e$
	服务器		$6p+1e$
SAP	客户端	6500	4 次 RSA 加密运算和一次 RSA 签名运算
	服务器		4 次 RSA 加密运算和一次 RSA 签名运算

从表 15.2 可以看出，IDKISAP 比 SAP 可以节省大量的通信量，只占 SAP 通信量约 6%。仅就计算量而言，IDKISAP 似乎比 SAP 需要更多的计算时间，因为根据文献[82]的结果，当使用 1G PIII CPU 的计算机运算时，IDKISAP 协议中 Client 需耗时 51ms，服务器需耗时 59ms；而 SAP 协议中客户端和服务器均需耗时 20ms。然而，在上述分析中，没有考虑证书状态查询给 SAP 协议带来的时间延迟，这是因为采用不同的证书状态查询方式和不同的证书链长度都会影响该值，但不论使用 OCSP 协议在线查询还是通过下载 CRL 检查，在最优的情况下，客户端和服务器也至少各需要一轮网络交互。由于网格系统中往往包含许多信任域，而且认证活动具有动态性，因此很难通过预先下载 CRL 的方式降低实时交互次数。因此在实际应用时，IDKISAP 协议与 SAP 协议的耗时是基本相当的，甚至更低。

同 LP 方案[75]比较：LP 方案同该方案非常类似，均通过改造 SAP 协议实现双向认证，而且效率基本相当。但 LP 方案基于 HIDC 方案，而该方案基于 15.2 节的 IDKIE 方案和 IDKIS 方案。LP 方案中没有详细讨论对用户长期私钥的撤销方法。但如果采用 BF 撤销方案[5]，即通过在用户公钥信息中增加有效期的方法实现自动撤销，为了实现细粒度的密钥

撤销必然要缩短用户密钥的有效期,从而导致用户需要频繁向 PKG 申请私钥,而且 LP 方案中建议使用离线方法分发用户私钥,这样会极大地增加用户负担。而该方案使用的密钥撤销方法可以容易地实现密钥撤销,而且在用户密钥整个生命周期内,只需要一次安全分发过程。另外,该方案可以通过密钥更新信息的形式灵活地表达用户的角色信息,而 LP 方案中则不易做到这一点。

同 Mao 方案[78]比较:Mao 方案由于不需要通信双方交互就可实现认证,因此具有较高的效率。但该方案只能在单一信任域环境中应用,而且该方案中生成的是静态会话密钥,攻击者一旦获得该密钥就可以完全控制双方通信。该方案可以实现跨信任域的认证,而且认证双方每次会话都可以重新协商会话密钥,攻击者即使获得某次会话的会话密钥,也只能获得当次会话内容。

15.5 小结

由于其独特性质,ID-PKI 在很多方面比传统的基于证书的 PKI 更具优势,其应用前景被广泛看好。然而就目前而言,ID-PKI 技术还远未成熟,仍存在许多问题亟待解决。信息安全国家重点实验室 PKI 项目组针对 ID-PKI 中的一些核心问题进行了深入系统研究[82~85],并研制了一套 ID-PKI 原型系统,部分工作已在文献[33]中做了较为详细的总结。本章主要介绍了我们自己的一些工作,对其他一些有代表性的工作也做了一些综述和对比介绍,感兴趣的读者可从文中介绍的线索参阅相关文献。路晓明博士也参加了本章写作,作者在此表示衷心的感谢。

参 考 文 献

[1] Shamir A. Identity-based cryptosystems and signature schemes. In Advances in Cryptology-Crypto1984, LNCS 196, 47~53, 1984. Springer.

[2] Fiat A, Shamir A. How to prove yourself: practical solutions to identification and signature problems. In Advances in Cryptology-Crypto1986, LNCS 263, 186~194, 1986. Springer-Verlag.

[3] Guillou L C, Quisquater JJ. A "paradoxical" identity-based signature scheme resulting from zero-knowledge. In Advances in Cryptology-Crypto1988, LNCS 403, 216~231, 1988. Springer-Verlag.

[4] Cocks C. Anidentity based encryption scheme based on quadratic residues. In the 8th IMA International Conference on Cryptography and Coding, LNCS 2260, 360~363, 2001. Springer-Verlag.

[5] Boneh D, Franklin M. Identity-based encryption from the weil pairing. In Advances in Cryptology-Crypto 2001, LNCS 2139, 213~219, 2001. Springer-Berlag.

[6] Linn J, Branchaud M. An examination of asserted PKI issues and proposed alternatives. In the 3rd Annual PKI R&D Workshop, 2004.

[7] Housley R, Ford W, Polk W. RFC 2459: Internet X. 509 public key infrastructure certificate and CRL profile. 1999.

[8] Malpani A, Galperin S, Myers M. RFC 2560: X. 509 internet public key infrastructure online certificate status protocol-OCSP. 1999.

[9] Sakai R, Ohgishi K, Kasahara M. Cryptosystems based on pairing. In Sympoium on Cryptography and Information Security-SCIS 2000, 26～28, 2000.
[10] Bellare M, Namprempre C, Neven G. Security proofs for identity-based identification and signature schemes. In Advances in Cryptology-EuroCrypt 2004, LNCS 3027, 268～286, 2004. Springer-Verlag.
[11] Libert B, Quisquater JJ. The exact security of an identity based signature and its applications. 2004, Cryptology ePrint Archive, Report 2004/102.
[12] Boneh D, Boyen X. Efficient selective-ID secure identity based encryption without random oracles. In Advance in Cryptology-EuroCrypt 2004, LNCS 3027, 223～238, 2004. Springer-Verlag.
[13] Sakai R,Kasahara M. ID based cryptosystems with pairing on elliptic curve. 2003, Cryptology ePrint Archive, Report 2003/054.
[14] Waters B. Efficient identity-based encryption without random oracles. In Advances in Cryptology-EuroCrypt 2005, LNCS 3494, 440～456, 2005. Springer-Verlag.
[15] Cha J C, Cheon J H. An identity-based signature from gap diffie-hellman groups. In Practice and Theory in Public Key Cryptography-PKC2003, LNCS 2567, 18～30, 2003. Springer-Verlag.
[16] Hess F. Efficient identity based signature schemes based on pairings. In Selected Areas in Cryptography-SAC 2002, LNCS 2595, 310～324, 2002. Springer-Verlag.
[17] Gentry C,Silverberg A. Hierarchical ID-based cryptography. In Advances in Cryptology -AsiaCrypt 2002, LNCS 2501, 548～566, 2002. Springer-Verlag.
[18] Boyen X. Multipurpose identity-based signcryption: a swiss army knife for identity-based cryptography. In Advances in Cryptology-Crypto 2003, LNCS 2729, 382～398, 2003. Springer-Verlag.
[19] Libert B, Quisquater JJ. New identity based signcryption schemes based on pairings. 2003, Cryptology ePrint Archive, Report 2003/023.
[20] Malone-Lee J. Identity-based signcryption. 2002, Cryptology ePrint Archive, Report 2002/098.
[21] McCullagh N, Barreto P S L M. Efficient and forward-secure identity-based signcryption. 2004, Cryptology ePrint Archive, Report 2004/117.
[22] Chen L,Kudla C. Identity based authenticated key agreement protocols from pairings. In the 16th IEEE Computer Security Foundations Workshop-CSFW 2003, 219～233, 2003. IEEE Computer Society Press.
[23] McCullagh,N, Barreto P S L M. A new two-party identity-based authenticated key agreement. 2004, Cryptology ePrint Archive, Report 2004/117.
[24] Smart N P. An identity based authenticated key agreement protocol based on the Weil pairing. Electronics Letters, 2002, 38: 630～632.
[25] Wang Y. Efficient identity-based and authenticated key agreement protocol. 2005, Cryptology ePrint Archive, Report 2005/108.
[26] Dutta R, Barua R, Sarkar P. Pairing-based cryptography: a survey. 2004, Cryptology ePrint Archive, Report 2004/064.
[27] Gagnee M. Identity-based encryption: a survey. RSA Laboratories Cryptobytes, 2003, 6(1): 10～19.
[28] Gorantla M C, Gangishetti R, Saxena A. A survey on ID-based cryptographic primitives. 2005, Cryptology ePrint Archive, Report 2005/094.
[29] Fujisaki E,Okamoto T. Secure integration of asymmetric and symmetric encryption schemes. In Advance in Cryptography-Crypto 1999, LNCS 1666, 537～554, 1999. Springer-Verlag.

[30] Bellare M, Desai A, Pointcheval D. Relations among notions of security for public-key encryption schemes. In Advances in Cryptology-Crypto 1998, LNCS 1462, 26～45, 1998. Springer-Verlag.

[31] Rackoff C, Simon D. Noninteractive zero-knowledge proof of knowledge and chosen ciphertext attack. In Advances in Cryptology-Crypto 1991, LNCS 547, 433～444, 1991. Springer-Verlag.

[32] Pointcheval D, Stern J. Security arguments for digital signatures and blind signatures. Journal of Cryptology, 2000, 13(3): 361～396.

[33] 路晓明. 基于身份的 PKI 系统研究, 中国科学院软件研究所博士学位论文, 2006.

[34] Dodis Y, Katz J, Xu S. Key-insulated public key cryptosystems. In Advances in Cryptology-EuroCrypt 2002, LNCS 2332, 65～82, 2002. Springer-Verlag.

[35] Shamir A. How to share a secret. Communications of the ACM, 1979, 22(11): 612～613.

[36] Feldman P. A practical scheme for non-interactive verifiable secret sharing. In the 28th IEEE symposium on Foundations of Computer Science, 427～437, 1987.

[37] Desmedt Y. Some recent research aspects of threshold cryptography. In the 1st International Information Security Workshop, 158～173, 1997.

[38] Desmedt Y, Frankel Y. Threshold cryptosystem. In Advances in Cryptology- Crypto1989, LNCS 435, 307～315, 1989. Springer-Verlag.

[39] Santis A D, Desmedt Y, Frankel Y. How to share a function securely. In the 26th annual ACM symposium on Theory of computing, 522～533, 1994. ACM Press.

[40] Backes M, Cachin C, Strobl R. Proactive secure message transmission in asynchronous networks. In the 22nd ACM Symposium on Principles of Distributed Computing- PODC 2003, 223～232, 2003. ACM Press.

[41] Barak B, Herzberg A, Naor D, Shai E. The proactive security toolkit and applications. In the 5th ACM Conference on Computer and Communications Security, 18～27, 1999. ACM Press.

[42] Herzberg A, Jarecki S, Krawczyk H, Yung M. Proactive secret sharing or: How to cope with perpetual leakage. In Advances in Cryptology-Crypto1995, LNCS 963, 339～352, 1995. Springer-Verlag.

[43] Ostrovsky R, Yung M. How to withstand mobile virus attacks. In the 10th Annual ACM Symposium on Principles of Distributed Computing -PODC 2003, 51～59, 1991. ACM Press.

[44] Bellare M, Miner S. A forward-secure digital signature scheme. In Advances in Cryptology-Crypto 1999, LNCS 1666, 431～448, 1999. Springer-Verlag.

[45] Canetti R, Halevi S, Katz J. A forward-secure public-key encryption scheme, 2003. Cryptology ePrint Archive, Report 2003/083.

[46] Bellare M, Palacio A. Protecting against key exposure: strongly key-insulated encryption with optimal threshold. 2002, Cryptology ePrint Archive, Report 2002/064.

[47] Dodis Y, Katz J, Xu S, Yung M. Strong key-insulated signature schemes. In Practice and Theory in Public Key Cryptography-PKC 2003, LNCS 2567, 130～144, 2003. Springer.

[48] Dodis Y, Franklin M K, Katz J. A generic construction for intrusion-resilient public-key encryption. In RSA-Cryptographers' Track 2004, LNCS 2964, 81～98, 2004. Springer.

[49] Itkis G, Reyzin L. SiBIR: Signer-Base Intrusion-Resilient Signatures. In Advances in Cryptology-Crypto 2002, LNCS 2442, 499～514, 2002. Springer-Verlag.

[50] Anderson R. Two remarks on public key cryptology. 2002, Technical Report UCAM-CL-TR-549, University of Cambridge, Computer Laboratory.

[51] Bellare M, Rogaway P. Entity authentication and key distribution. In Advances in Cryptology-

Crypto 1993, LNCS 773, 110～125, 1993. Springer-Verlag.

[52] Blake-Wilson S, Johnson D, Menezes A. Key agreement protocols and their security analysis. In the 6th IMA International Conference on Cryptography and Coding, LNCS 1355, 30～45, 1997. Springer-Verlag.

[53] Canetti R, Krawczyk H. Analysis of key-exchange protocols and their use for building secure channels. In Advances in Cryptology-EuroCrypt 2001, LNCS 2045, 453～474, 2001. Springer-Verlag.

[54] Boneh D, Ding X, Tsudik G, Wong C. A method for fast revocation of public key certificates and security capabilities. In the 10th USENIX Security Symposium, USENIX, 2001.

[55] Boneh D, Ding X, Tsudik G. Identity based encryption using mediated RSA. In the 3rd workshop on Information Security Application, 2002.

[56] Ding X, Tsudik G. Simple identity-based cryptography with mediated RSA. In RSA-Cryptographers' Track 2003, LNCS 2612, 193～210, 2003. Springer.

[57] Libert B, Quisquater J J. Efficient revocation and threshold pairing based cryptosystems. In Symposium on Principles of Distributed Computing-PODC 2003, 163～171, 2003.

[58] Baek J, Zheng Y. Identity-based threshold decryption. In Practice and Theory in Public Key Cryptography-PKC 2004, LNCS 2947, 248～261, 2004. Springer-Verlag.

[59] Hanaoka Y, Hanaoka G, Shikata J, Imai H. Identity-based hierarchical strongly key-insulated encryption and its application. 2004, Cryptology ePrint Archive, Report 2004/338.

[60] Chen L, Harrison K, Soldera D, Smart N. Applications of multiple trust authorities in pairing based cryptosystems. In Infrastructure Security-InfraSec 2002, LNCS 2437, 260～275, 2002. Springer-Verlag.

[61] Al-Riyami S, Paterson K G. Certificateless public key cryptography. In Advances in Cryptology-AsiaCrypt 2003, LNCS 2894, 452～473, 2003. Springer-Verlag.

[62] Cheng Z, Comley R, Vasiu L. Remove key escrow from the identity-based encryption system. In IFIP TCS 2004, 37～50, 2004.

[63] Lee B, Boyd C, Dawson E, Kim K, Yang J, Yoo S. Secure key issuing in ID-based cryptography. In Australasian Information Security Workshop-AISW 2004, 69～74, 2004.

[64] Gangishetti R, Gorantla M C, Das M L. An efficient secure key issuing protocol in ID-based cryptosystems. In the International Conference on Information Technology: Coding and Computing-ITCC 2005, 1, 674～678, 2005.

[65] Oh J, Lee K, Moon S. How to solve key escrow and identity revocation in identity-based encryption schemes. In International Conference on Information Systems Security-ICISS 2005, LNCS 3803, 290～303, 2005. Springer-Verlag.

[66] Xu C, Zhou J, Qin Z. A note on secure key issuing in ID-based cryptography. 2005, Cryptology ePrint Archive: Report 2005/180.

[67] Gangishetti R, Gorantla M C, Das M L, Saxena A. Cryptanalysis of key issuing protocols in iD-based cryptosystems. 2005, http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:cs/0506015.

[68] Sui A F, Chow S S M, Hui L C K, Yiu S M, Chow K P, Tsang W W, Chong C F, Pun K H, Chan H W. Separable and anonymous identity-based key issuing. In the 1st International Workshop on Security in Networks and Distributed Systems-SNDS 2005, 275～279, 2005. IEEE Computer Society Press.

[69] Butler R, Welch V, Engert D, Foster I, Tuecke S, Volmer J, Kesselman C. A national-scale authentication infrastructure. IEEE Computer, 2000, 33(12): 60～66.

[70] Foster I, Kesselman C, Tsudik G, Tuecke S. A security architecture for computational grids. In the 5th ACM conference on Computer and communications security, 83～92, 1998. ACM press.

[71] Nagaratnam N, Janson P, Dayka J, Nadalin A, Siebenlist F, Welch V, Foster I, Tuecke S. The security architecture for open GRID services. In Open GRID Service Architecture Security Working Group (OGSA-SEC-WG), 2002.

[72] Thompson M, Essiari A, Mudumbai S. Certificate-based authorization policy in a PKI environment. ACM Transactions on Information and System Security, 2003, 6(4): 566～588.

[73] Gutmann P. PKI: it's not dead, just resting. IEEE Computer, 2002, 35(8): 41～49.

[74] Chen L, Lim H W, Mao W. User-friendly grid security architecture and protocols. In the 13th International Workshop on Security Protocols, 2005.

[75] Lim H W, Paterson K G. Identity-based cryptography for grid security. In the 1st IEEE International Conference on e-Science and Grid Computing-e-Science 2005, 395～404, 2005. IEEE Computer Society Press.

[76] Lim H W, Robshaw M J B. On identity-based cryptography and GRID computing. In the 4th International Conference on Computational Science-ICCS 2004, LNCS 3036, 474～477, 2004. Springer-Verlag.

[77] Lim H W, Robshaw M J B. A dynamic key infrastructure for GRID. In the European Grid Conference-EGC 2005, LNCS 3470, 255-264, 2005. Springer-Verlag.

[78] Mao W. An identity-based non-interactive authentication framework for computational grids. Hewlett-Packard Laboratories, technical report HPL-2004-096, 2004.

[79] Sandhu R S, Coyne E J, Feinstein. H. L. Role-based access control models. IEEE Computer, 1996, 29(2): 38～47.

[80] Qiang W, Jin H, Shi X, et al. RB-GACA: A RBAC based grid access control architecture. In Grid and Cooperative Computing: Second International Workshop-GCC 2003, LNCS 3032, 487～494, 2003. Springer-Verlag.

[81] Chen Y, Yang S, Guo L. Dynamic-role based access control framework across multi-domains in grid environment. In Grid and Cooperative Computing-GCC 2005, LNCS 3795, 161～165, 2005. Springer-Verlag.

[82] 路晓明,冯登国. 一种基于身份的多信任域网格认证模型. 电子学报, 2006, 34(4):577～582.

[83] Zhang Z F, Wong D S, Xu J, Feng D G. Certificateless Public-Key Signature: Security Model and Efficient Construction. ACNS 2006: 293～308.

[84] Lu X M, Feng D G. Security Proof of the Original SOK-IBS Scheme. I. J. Network Security 5(2): 176～181(2007).

[85] Zhang F, Feng D G. Identity-based PKI Scheme for Machine Readable Travel Document. AINA (2) 2006: 461～465.

第 16 章　可信计算平台远程证明协议

计算机安全问题由来已久，早期因计算资源昂贵，强调效率优先原则，所以安全问题考虑得比较少，尤其在体系结构上，没有建立相应的安全支撑机制。随着计算机应用尤其是网络应用的普及，安全问题也越来越突出。导致这种局面的根本原因在于：本地/网络应用程序代码无法保障代码自身的可信性，传统的各种基于软件的安全防御技术屡屡被攻击者攻陷。正是在这种背景下，可信计算平台技术应运而生，目前已成为计算机安全技术最主要的发展趋势之一，其基本思想是在计算平台上引入一个硬件信任根，通过这个硬件信任根构建主机的可信执行环境。

国际可信计算组织(TCG)制定了可信计算平台、可信存储和可信网络的一系列工业标准[1]，其方法是在主机平台、移动平台和嵌入式平台上嵌入专用的安全芯片 TPM(Trusted Platform Module)[2]，以此为硬件信任根解决可信计算平台信任的建立和证明问题。在国家有关主管部门和中国可信计算工作组(TCMU)的积极推动下，我国也制定了具有自主知识产权的 TCM(Trust Cryptographic Module)相关标准[3]，国内 IT 厂商相继研制出支持 TCM 标准的安全芯片及相关软、硬件产品，并积极推动示范应用。

远程证明是可信计算平台提供的一大特色和核心功能，是目前可信计算领域最为热门的研究方向之一。TCG 的两个版本中提出了两种远程证明方案：基于可信第三方 Privacy-CA 的证明方案和直接匿名证明(DAA)方案，它们确立了远程证明的基本原则。针对这两个方案的不足，人们展开了许多有益的探索并取得了多项研究成果。

远程证明包含两层含义：一是 TPM/TCM 身份的证明，也称之为平台身份的证明，因为 TPM/TCM 可以唯一标识一个平台的身份，TPM/TCM 身份证明就是要提供一个只有 TPM/TCM 才知道的证据，通过 TPM/TCM 的身份密钥 AIK 对内部数据进行签名实现；二是平台证明，它是提供证据证明一个平台完整性值集合的证明过程，这是通过用 TPM/TCM 中的 AIK 对一个平台配置寄存器(PCR)集合进行数字签名来实现的。

本章结合国际上的一些现有成果，重点介绍我们在可信计算平台远程证明方面的研究成果。

16.1　多远程证明实例动态更新证明方案

TCG 框架下的远程证明方案得到了国内、外众多学者、研究机构的广泛关注，众多的研究成果中较为典型的有 IBM 研究院提出的完整性度量框架 IMA[4][5]。这些远程证明方案中，有的是基于软件的，有的是基于语义的，有的是基于属性的。基于属性的远程证明方案具有明显的发展优势，它可以更好地解决远程证明的隐私泄露和滥用证明结果等问题，并可降低证明的复杂性。IBM 研究院在文献[6]中引入可信第三方转换属性，提出了基于属性的远程证明框架，随后在文献[7]中给出了基于属性的远程证明方案的软、硬件实现方法，文献[8]中提出了基于属性的远程证明协议(简称 PBA 协议)，文献[9]中在可信引导器

TrustedGrub 的基础上，对基于属性的远程证明方法、属性验证和撤销等实现技术进行了研究，这些研究不断地推动了基于属性的远程证明方案的发展。这些工作改进了 TCG 远程证明方案的各方面不足之处，扩展了远程证明方法，但对于远程证明的动态性和并发性研究很少。文献[10]讨论了计算环境配置改变和封装数据不可用的问题，提出了基于属性封装的解决方法，实际上远程证明也存在同样的问题。当系统软、硬件配置和状态发生改变时，原有的远程证明将无效，必须再次动态地进行更新证明，这就涉及远程证明的动态性。可信计算平台远程信任建立实际应用过程中，系统中可能存在着多个远程证明会话实例(Multi-RAI，Multiple Remote Attestation Instance)，这就要求 Multi-RAI 能够并发证明不同的 RAI 关联运行环境的可信性。与单个远程证明应用相比，又存在许多新的复杂问题。

解决现有网络安全最常用的安全协议有 SSL/TLS 协议和 IPSec 协议，SSL/TLS 协议和 IPSec 协议认证终端用户身份，保证网络通信数据的机密性和完整性，而远程证明保证终端运行环境的可信性，将这二者结合可以极大地增强网络应用的安全性，TCG 的体系结构互操作规范[11]和 TNC 规范[12]都涉及相关问题的讨论。文献[13]中提出了使用远程证明扩展 SSL 协议的方案，通信终端通过协商安全参数和在 SSL 协议上证明平台配置，以此达到建立远程可信通道的目标。TCG 工作组也正着力于建立 TLS 协议上的远程证明扩展规范[14]。Multi-RAI 安全连接的讨论也是建立在 TLS 协议扩展的远程证明基础之上。

系统环境复杂多变，需要采用多种安全机制来保护和标识软件运行环境，虚拟技术[15]提供的强隔离机制为保护和维护计算环境的完整性提供了良好的基础。无论是以 Xen[16]为代表的虚拟技术、微内核技术[17]、Intel 的 LT 技术[18]，还是 Microsoft 的 NGSCB 技术[19]，都是在可信虚拟机监控器上将系统划分为不同安全级的计算隔间(Compartment)，严格限制 Compartment 之间的信息流，以达到控制和维护计算环境安全性的目的。Multi-RAI 证明的可信计算环境是以 Compartment 为基本单元，证明 Compartment 中组件构建的运行环境是否可信。

系统打补丁、应用程序的升级，乃至恶意程序的篡改等致使可信计算平台的状态发生改变，状态的改变导致必须进行状态更新证明。Multi-RAI 的并发性和平台状态的动态性，都极大地增加了远程证明的复杂度。TCG 规范提出的远程证明方法主要针对平台静态状态的远程证明，无法解决平台状态动态改变的远程证明。Multi-RAI 证明最直观的解决方法便是扩展单个 RAI 证明，但这种简单扩展会引发许多新问题，图 16.1 归纳了单个 RAI 证明扩展到 Multi-RAI 证明的常见问题。

图 16.1(a)中描述的是非一致性证明，在第(4)步应该证明的 RAI_A 计算环境状态 β，可是证明时 RAI_B 改变 Compartment 状态为 γ，RAI_A 实际证明的状态为 γ，证明状态不一致，这是由于 Multi-RAI 计算环境状态相互影响造成了证明状态不一致。TCG 规范采用服务器发送新鲜值 Nonce，防止远程证明的重放攻击，但是这种方法在 Multi-RAI 的更新证明中将不再适用，除首次证明外，更新证明都是在 Requester 不知道的情形下发生，如果通知 Requester 发送 Nonce，Compartment 此时可能已经改变为另一状态，这将出现如图 16.1(a)所示的非一致性证明。如图 16.1(b)所描述的是 RAI 证明的重放攻击证明，第(4)步 RAI_A 计算环境状态已经改变为 γ，而不诚实的用户可以重放状态 β 的证明。Multi-RAI 可信计算环境对于远程证明提出了新的安全要求。

(1) 证明的动态性。可信计算环境的软、硬件配置和状态动态发生改变时，RAI 必须动

(1) PLATFORM: 可信状态 α
(2) (i) REQUESTER → RAI_A: Nonce值 n_1
(ii) RAI_A → REQUESTER：Attest(α,n_1)
(3) PLATFORM: α → β, RAI_A 状态改变
(4) (i) REQUESTER → RAI_A: Nonce值 n_2
(ii) PLATFORM: β → γ, RAI_B 状态改变
(iii) RAI_A → REQUESTER：Attest(γ,n_2)

(a)

(1) PLATFORM: 可信状态 α
(2) (i) REQUESTER → RAI_A: Nonce值 n_1
(ii) RAI_A → REQUESTER：Attest(α,n_1)
(3) (i) PLATFORM: α → β, RAI_A 改变状态
(ii) RAI_A → REQUESTER：UpdateAttest(β)
(4) (i) PLATFORM: β → γ, RAI_A 改变状态
(ii) RAI_A → REQUESTER：UpdateAttest(β)

(b)

图 16.1 Multi-RAI 证明的常见问题

态地向 Requester 证明更新后的新状态可信。

（2）证明的一致性。RAI_A 证明可信计算环境中与它关联的运行状态时，不能由于其他 RAI 改变了可信计算环境的状态，而导致 RAI_A 错误的证明改变后的新状态，RAI_A 的证明必须与 RAI_A 的关联状态一致。

（3）证明的并发性。Multi-RAI 环境下同时存在多个并发运行的向不同 Requester 证明计算环境的可信性的 RAI。不能因为 RAI 证明而锁定当前计算环境，亦即 RAI 证明时，不允许平台状态更新，不允许其他 RAI 同时进行证明。不管平台的运行环境经过了多少次改变，Multi-RAI 还能够并发证明其不同的最新关联状态。

（4）防重放攻击。RAI 的关联状态与 RAI 证明一一对应，但是当关联状态改变为不安全状态时，不诚实的用户或恶意程序能够重用原有的证明数据，使 Requester 相信 RAI 仍然可信，这就是 Multi-RAI 的重放攻击。

16.1.1 远程证明模型

1. TPM/TCM 基础

TPM/TCM 是一个安全芯片（实质上是一个密码芯片），可提供密码操作、完整性管理、密钥管理、安全存储、远程证明等安全功能。这里主要涉及 TPM/TCM 的完整性配置管理、单调计数器、TPM/TCM 身份密钥等功能。

TPM/TCM 内部包含一组不可篡改的平台配置寄存器（PCR），TPM/TCM 度量可信计算环境的软、硬件状态，将度量结果保存在这些 PCR 中。PCR 的主要操作包括扩展、重置、签名等，当系统启动后，PCR 不能重置和篡改，只能进行杂凑值的扩展，其扩展方法为 TPM_Extend(α,β)=$H(\alpha \| \beta)$（这里以 TPM 为例），其中 α 为 PCR 的旧值，β 为扩展值，输出 PCR 的新值，TCG 规范指定的 Hash 算法为 SHA1。令 $R=\{0,1\}^{160}$，将 PCR 扩展操作定义为点积（“·”）运算，则 $R\times R\rightarrow R:(\alpha,\beta)\mapsto H(\alpha \| \beta)=\alpha\cdot\beta$。

以可信计算平台的初始环境创建过程为例，主机平台的 BIOS 杂凑运算的度量结果为 β，OS Loader 为 λ，OS kernel 为 κ，OS 加载的内核模块（如特定硬件的驱动、内核服务模块等）度量结果为 $\chi_1,\cdots,\chi_m$，加载的用户空间的应用程序为 $\chi_{m+1},\cdots,\chi_{m+n}$，那么对整个信任链进行的扩展操作是

$$\begin{aligned} \mathrm{PCR} &= H(H(\cdots H(H(H(H(0^{160} \| \beta) \| \lambda) \| \kappa) \| \chi_1) \| \cdots) \| \chi_{m+n}) \\ &= 0^{160} \cdot \beta \cdot \lambda \cdot \kappa \cdot \chi_1 \cdot \cdots \cdot \chi_{m+n} \end{aligned} \tag{16-1}$$

TPM/TCM 单调计数器(Monotonic Counter)值保存在 TPM/TCM 内部的非易失(NV)存储区域内,且计数器只能够被增加。创建一个新计数器需要 Owner 的授权,增加一个计数器值需要计数器相关授权口令,删除计数器要么需要 Owner 授权,要么需要计数器相关的授权口令。创建一个新计数器后,计数器被赋予了一个初始值,通常初始值不从零开始。计数器可用来标识可信计算环境状态的更新顺序和防止证明数据的重放攻击。

2. 体系结构

组件通过度量变量衡量其安全属性,我们为组件颁发属性证书,验证组件是否可信,图 16.2(a)是以组件为证明单位的体系结构,系统包含组件生产厂商、用户(简记为 U)、证书发布权威(T)。组件生产厂商生产和发布组件,提供组件度量类别和度量变量;证书发布权威机构评估组件安全属性,发布、验证和撤销组件属性证书;用户平台由主机系统和 TPM/TCM 安全模块组成,组件属性证明和可信计算环境的更新证明发生在用户平台安全通信过程中。

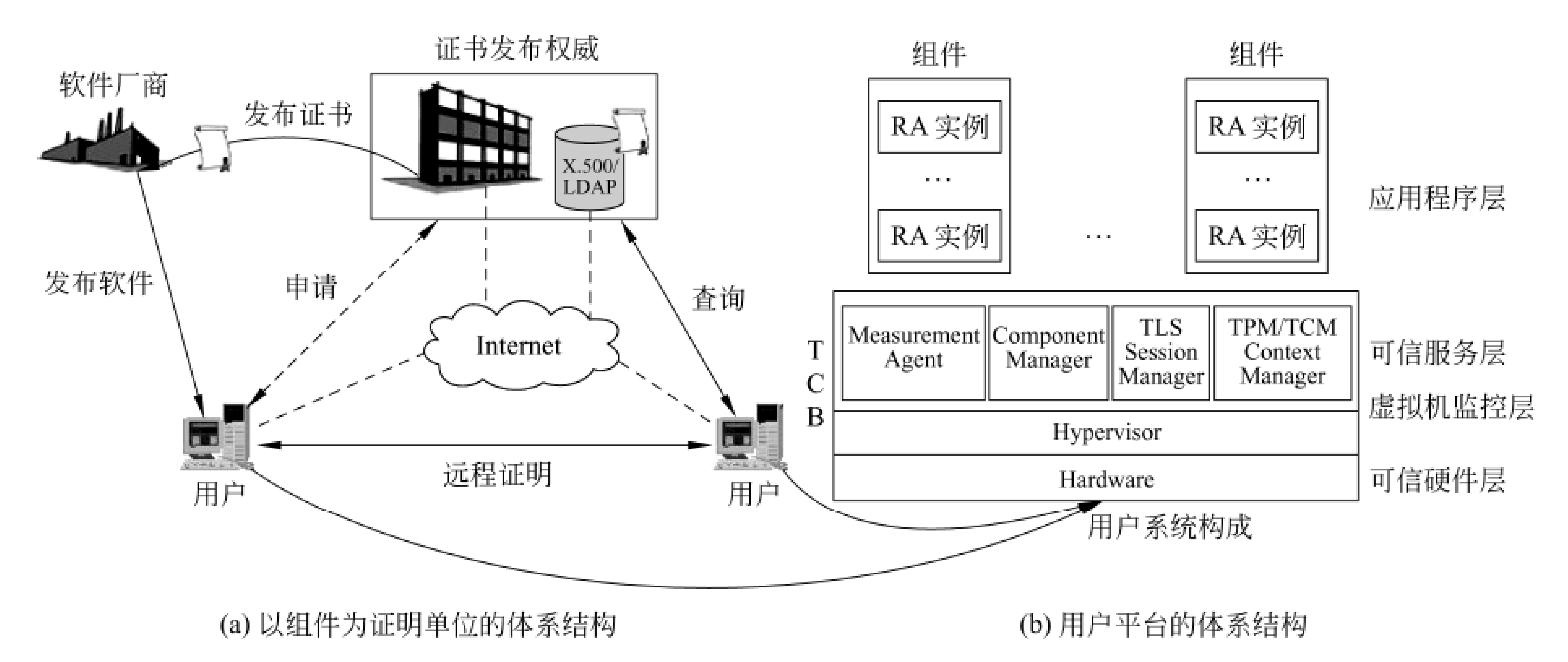


图 16.2 可信计算环境证明模型和体系结构

设 T 的公私钥为($\mathrm{SK_T}$,$\mathrm{PK_T}$),经过 T 评估认证的组件安全属性集合为 P,P 包含有长度为 l 的代表不同组件安全属性的字符串,具体形式为

$$P = \{p_1, p_2, \cdots, p_n\} \subset \{0,1\}^l \backslash \{0^l\}$$

组件用三元组(id,χ,p)表示,id 是组件的身份标识,χ 是组件全部度量变量按照组件度量算法得到的度量值,p 是组件经过 T 评估满足的安全属性,$p \in P$。T 为组件(id,χ,p)颁发的属性证书,记为

$$\mathrm{cert}(T, \mathrm{id}, \chi, p) = (\mathrm{id}, \chi, p, \mathrm{Sign}(\mathrm{SK_T}, (\mathrm{id}, \chi, p))) \tag{16-2}$$

为了便于组件属性撤销查询,T 周期性地颁发组件属性撤销列表 CRL,同时提供在线证书查询服务 OCSP 等。

图 16.2(b)是用户平台的体系结构,可信服务层、虚拟机监控层、可信硬件层共同构成了用户可信计算平台的可信计算基(TCB),TCB 的上面是运行各个相互隔离的隔间(Compartment),Compartment 可以采用 Xen 提供的隔离虚拟机,或 NGSCB 提供的保护分

区，或微内核系统实现，我们的系统使用 Xen 可信平台上的虚拟机实现 Compartment。远程证明时，首先证明 TCB 的可信，然后证明 Compartment 中 RAI 实例关联的组件安全属性。可信服务层提供了组件证明必需的存储、会话、度量管理等服务。

(1) MA(Measurement Agent)。进行平台组件的完整性度量，组件更新时对被更新的组件重新度量。

(2) CM(Component Manager)。管理系统中各个组件的安装注册、删除卸载和更新升级等，保存有注册组件的组件属性证书。CM 负责处理 Compartment 中组件的更新请求，如果组件的安全属性发生变化，则触发 MA 对组件重新度量，确保组件满足属性证书中所声明的安全属性。CM 用组件列表来保存组件注册信息和组件属性证书信息，多个 Compartment 拥有相同的组件，CM 共享该组件列表项，若组件发生更新，CM 使用 COW (Copy On Write)机制修改组件列表项。

(3) TSMG(TLS Session Manager)。管理系统中全部远程证明 TLS 会话实例 RA，每个远程证明实例 $RAI\in RA$ 总是绑定一个会话关联组件集合 C，这些会话关联组件为叶节点组成一棵会话组件树。Compartment 中对任意的 $c\in C$，如果组件 c 发生更新，CM 会检查更新后的组件 c 的安全属性是否发生改变，如果改变则通知 TSMG 进行更新证明，验证成功则重用原有的 SSL 会话，但使用新的会话密钥。

(4) TCMG(TPM/TCM Context Manager)。TPM/TCM 的上下文管理负责维护各个 Compartment 的 TPM/TCM 操作环境，包括 Compartment 关联的密钥、会话、数据，以及各个 Compartment 绑定的物理 PCR，各个 RAI 关联的 VPCR(Virtual PCR)。Compartment 每创建一个 RAI 时，TCMG 在相应的 Compartment 的上下文中创建一个 VPCR 用来描述 RAI 关联的组件配置和状态，VPCR 值为组件会话树的根节点扩展物理 PCR 后的值。

3. 证明流程概述

图 16.3 描述了可信计算环境证明的生命周期过程。可信计算平台每创建一个

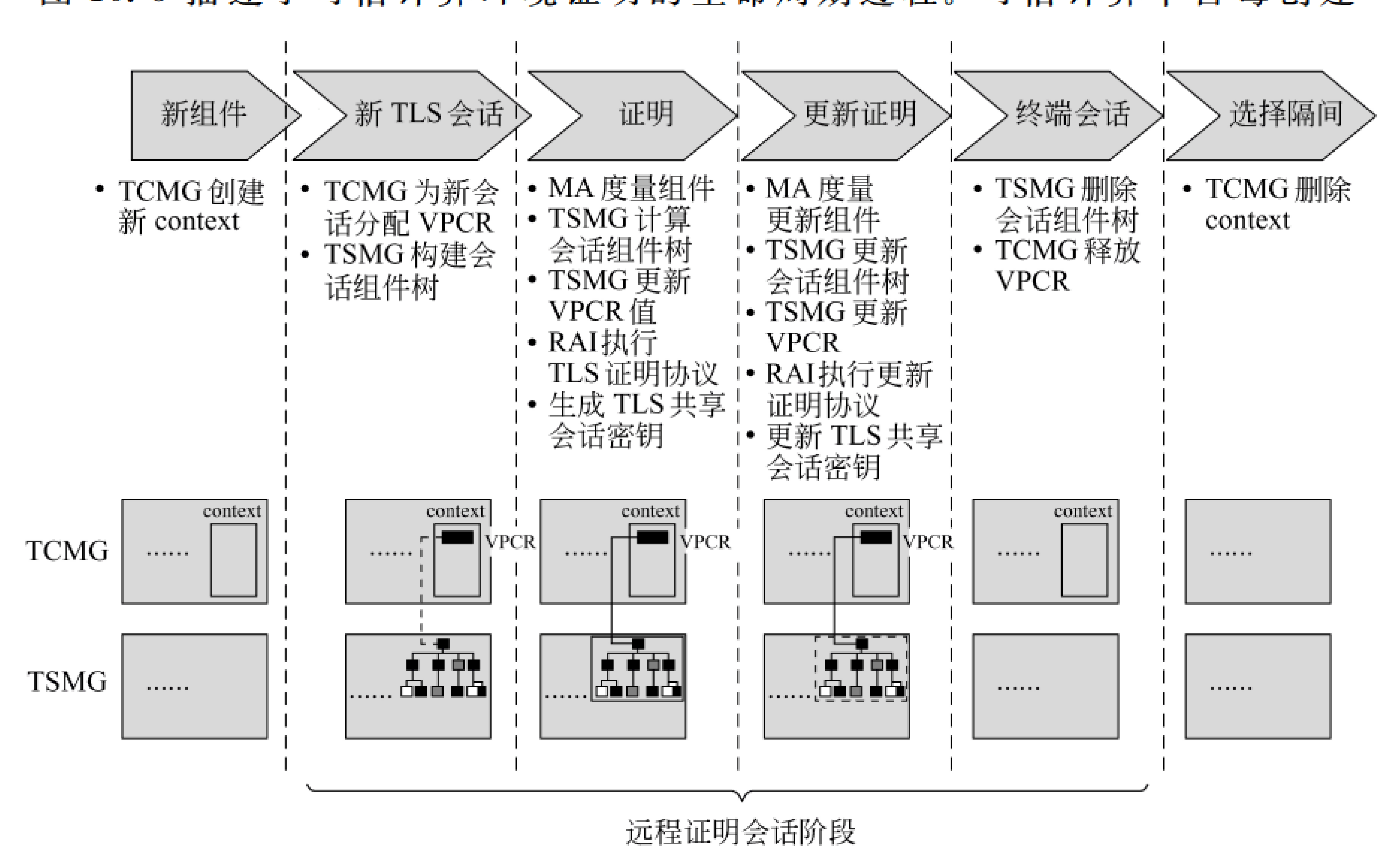


图 16.3　Multi-RAI 证明生命周期

Compartment 时，TCMG 就为该 Compartment 创建一个 TPM/TCM 上下文(Context)。当发起一个新的 TLS 会话时，TCMG 需要为该会话分配保存组件配置和状态的 VPCR，TSMG 则建立与该会话安全相关的会话组件树，会话组件树的组件用来衡量 RAI 会话是否满足某种安全属性。当一个 RAI 会话实例执行时，将触发 MA 对会话关联组件进行度量，度量完成后 TSMG 将计算会话组件树，然后 TPM/TCM 对证明数据签名，最后 RAI 运行扩展了远程证明的 TLS 协议完成证明。

可信计算环境的证明周期中，Multi-RAI 证明主要包括以下 3 个步骤。

(1) 组件度量。由 MA 执行的对组件的度量变量进行度量和评估的算法。MA 度量的组件应在 CM 保存的组件属性证书包含的度量变量范围之内，MA 完成度量后使用组件度量属性证书验证组件配置状态，如果二者不符，则由 CM 通知系统管理员组件已经不符合组件属性证书声明的安全属性。

(2) 会话组件树计算。RAI 会话要求满足安全属性 p，CM 将会话安全属性映射为平台组件安全属性，如 $p \Rightarrow \bigwedge_{i=1}^{n} p_i = p_1 \wedge \cdots \wedge p_n$，$p_i$ 是组件 c_i 的安全属性。TSMG 为这些与会话相关的组件创建组件树，被度量的组件经过验证后，将用组件度量值(或属性值)扩展计算会话组件树。

(3) Multi-RAI 证明。包括 RAI 证明和 RAI 更新证明。

RAI 证明是首次建立远程证明网络通信时必须进行计算环境可信性的证明。RAI 会话实例证明协议建立在 TLS(SSL)协议的基础上，在证明用户身份、密钥协商的同时，证明组件安全属性和 TPM/TCM 平台身份。

RAI 更新证明发生在 RAI 证明之后，是对可信计算环境改变后的状态进行证明。无论是用户合法的组件更新，还是恶意攻击导致组件配置和状态的改变，都会导致可信计算环境变化，此时将无法保障原有 RAI 会话运行环境是否可信，组件更新时 CM 将驱动 MA 重新度量更新组件，TSMG 进行更新组件证明。

16.1.2 远程证明方案

16.1.1 小节描述了可信计算环境证明总体过程，本节将提供具体的实现算法和协议，来保证通信双方的计算环境可信。可信计算环境证明方案 $\Gamma = (G, M, A, A_V, U, U_V)$ 是由一系列概率 Oracle[20] 算法构成。这些算法运行在组件度量、会话组件树计算、Multi-RAI 证明等主要步骤中，其中密钥生成算法 G 是由 TPM/TCM 芯片完成，度量算法 M 是由度量代理 MA 完成，证明算法 A 和更新证明算法 U 是由证明平台的 TSMG 和 TCMG 协同执行，验证算法 A_V 和 U_V 是由证明请求者 Requester 执行。令 I 是 Compartment 标识(Identities)的集合，A_i 是 Compartment $i(i \in I)$ 的 RAI 集合，$I \subseteq \{(i, ra): i \in I, ra \in A_i\}$。

(1) G 是 TPM/TCM 密钥生成算法，生成 Multi-RAI 证明所需要的平台身份密钥对 AIK(TCM 中记为 PIK)。给定安全参数 k，TPM/TCM 计算出密钥对 $(\mathrm{sk}, \mathrm{pk}) \leftarrow G(1^k)$。更为可靠的方法是为每个 Compartment 生成由 AIK 认证，带有 SKAE 扩展[21]的远程证明密钥(RAK)，专门用于 Multi-RAI 证明。

(2) M 是组件度量算法，输入组件 id，MA 就完成相应的组件度量 $M(\mathrm{id}) := \{\log, \chi\}$。

(3) A 是 RAI 的证明概率算法，亦即建立 RAI 会话时首次向 Requester 证明的算法。

算法执行时，TCMG 生成证明凭证：$s \leftarrow A(i,a,D,t)$，$(i,a) \in I$，D 是 RAI[a]在 TPM/TCM 单调计数器为 t 时的证明状态数据，D 在系统实现时包括组件会话树数据、VPCR 和度量日志数据。当证明状态数据更新($D \rightarrow D'$)，TCMG 也使用算法 A 生成更新凭证：$u \leftarrow A(i,a,D',t')$。

(4) U 是 RAI 更新证明概率算法，给定$(i,a) \in I$，算法执行输出更新证明证据：$f \leftarrow U(i,a,D_j,D_m)$，$D_j$、$D_m$ 分别是单调计数器为 t_j 和 t_m 时的证明状态数据，D_m 是 D_j 更新后的值。

(5) A_V 是 RAI 会话建立时的证明验证算法，给定证明凭证(或更新凭证)$s = A(i,a,D,t)$ 和公钥 pk，算法返回(VPCR，t)$\leftarrow A_V(s,i,a,\text{pk})$，VPCR 是 RAI[$a$]在证明状态数据为 D，TPM/TCM 单调计数器为 t 时的 PCR 值。若验证不通过，输出“reject”。

(6) U_V 是 RAI 更新证明的概率验证算法，给定更新证明证据 $f = U(i,a,D_j,D_m)$和公钥 pk，算法返回{(VPCR$_j$，t_j)，(VPCR$_m$，t_m)}$\leftarrow U_V(f,i,a,\text{pk})$，VPCR$_j$ 是 RAI[a]在证明状态数据为 D_j，单调计数器为 t_j 的 PCR 值，VPCR$_m$ 是证明状态数据 $D_j \rightarrow D_m$ 更新后，计数器为 t_m 时的 PCR 值。若验证不通过，输出“reject”。

1. 组件度量算法

首先对几个术语做一些解释。

(1) 组件(Component)。可信计算系统中可被度量的硬件和软件模块。可信计算系统是由硬件和各种软件构成，并具有一定功能和属性的系统组成实体。BIOS、BootLoader、OS 内核和 Firefox 浏览器等都是一个组件。组件本身可信，再加上组件之间的联系是可信的，就能保证系统配置和运行状态是可信的。

(2) 度量类别(Measurement Class)。衡量一个组件是否可信，其方法有很多，有验证运行代码是否签名、组件运行的程序是否含有恶意特征码等，而 TPM/TCM 主要以 fingerprint 的方式记录组件运行时的状态，来判定组件是否满足一定的安全条件。度量组件加载到内存中可执行文件镜像、组件程序代码段(Text Section)、组件调用的内核模块、组件运行时动态链接库等，用不同的类别来衡量组件运行时的状态，这些度量分类定义为度量类别。

(3) 度量变量(Measurement Variable)。某一个组件度量类别中，具体刻画组件当前运行状态的衡量点称之为度量变量。例如，浏览器使用的系统动态链接库类别中，SSL 动态链接库(/usr/lib/libssl3. so)就是一个度量变量。图 16.4 是原型系统中对于浏览器组件 Firefox 的度量结果，这里选择了下列 4 个度量类别，对于每个度量类别，系统的度量代理 MA 使用系统 Hook 函数自动选取度量变量。

① 可执行文件镜像：程序运行时，加载到进程空间的可执行文件的内存镜像。

② 组件动态链接库：组件直接依赖的动态链接库，随组件一起发布，当组件程序运行需要某个链接库，在它动态加载时对其进行度量。

③ 系统动态链接库：组件运行所依赖系统动态链接库，组件需要调用这些动态链接库，动态加载进行度量。

④ 系统内核模块：组件运行所依赖的内核模块，在组件使用时对它进行度量。

除了这些度量类别外，还可以从组件配置数据、组件代码段数据、组件运行时关键数据结构等度量类别来描述组件的可信运行状态。

(4) 组件更新(Component Update)。系统用户为组件打补丁、在线升级、重新安装新

Attestation Manager

File View Option Test Help

ID	Name	Measurement
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/librdf.so	F3A0B94D64F071E2C57FBF410A1D2E
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libremote...	0304C0C5F73DA399FE35767B24C5FC
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libsearchs...	D62480F43D57E01B2E739A5AB74130
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libsystem...	D0FEDA5B92788D0459D2A65D78444
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libtoolkit...	AA24BCD2E443B885A49264C39686ED
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libtxmgr.so	0956BA43C6E292A9E81D9CADBCE93E
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libuconv.so	536AA2C76F6A339723A5E9F431B6A9
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libwebbr...	DFB6525C1F9F515C2B25AC06A10772
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libwidget...	BB911A538F356359DDC2920F233460
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libxmlext...	C75DADE2912465B6AE9B796674A400
0xE18DDA67	/usr/lib/firefox-1.5.0.1/components/libxpconn...	FE603CFD3F06F3B8DBD173A5B54BC8
0xE18DDA67	/usr/lib/firefox-1.5.0.1/firefox-bin	615CEE7DF8AC4697C652CE3ECC3BE9
0xE18DDA67	/usr/lib/firefox-1.5.0.1/libgfxpsshar.so	05A528091BF9BFF792E356BEF72E9C4
0xE18DDA67	/usr/lib/firefox-1.5.0.1/libgkgfx.so	F325E90B3608D9D9F30F3B262BC5D9
0xE18DDA67	/usr/lib/firefox-1.5.0.1/libgtkxtbin.so	CBFBC34BD1A55E8F974266D92AFB85
0xE18DDA67	/usr/lib/firefox-1.5.0.1/libjsj.so	DD7CF0FA6F5EDEF1E25077A45A9FB3
0xE18DDA67	/usr/lib/firefox-1.5.0.1/libmozjs.so	E71E7C2D2DF4E71426E60C9063077B
0xE18DDA67	/usr/lib/firefox-1.5.0.1/libxpcom.so	2AD177B1896C18AE8BF64013578FB9
0xE18DDA67	/usr/lib/firefox-1.5.0.1/libxpcom_compat.so	8364BDD877CD0E22702DD8FB1887A1
0xE18DDA67	/usr/lib/firefox-1.5.0.1/libxpcom_core.so	F952B382BDC2531946CE85D38991A3
0xE18DDA67	/usr/lib/firefox-1.5.0.1/plugins/libnullplugin.so	CE06928F6703C67C694CF396FA53672
0xE18DDA67	/usr/lib/firefox-1.5.0.1/plugins/libunixprintpl...	736EBE4CFE5CAD6CC4AD8A330D7A7
0xE18DDA67	/usr/lib/gconv/EUC-CN.so	E9E51F767A7026DB8001BE1A88B093
0xE18DDA67	/usr/lib/gconv/ISO8859-1.so	D73C062D55456C113A9C74D6446C5
0xE18DDA67	/usr/lib/gconv/UTF-16.so	0A85F790D132F35A38BDDA02ACC728
0xE18DDA67	/usr/lib/gconv/libGB.so	F37B3A0397064610 8D96281C214BFF
0xE18DDA67	/usr/lib/gnome-vfs-2.0/modules/libfile.so	AC22D78419561C25A2667E82AC712E
0xE18DDA67	/usr/lib/gtk-2.0/2.4.0/engines/libclearlooks.so	49C02C7B5A6BBF870047CEAA838D53

图 16.4　度量变量和度量类别实例

版本软件等操作，导致组件度量变量的度量结果发生变化，这个过程称为组件更新。广义地讲，新安装组件、升级组件、删除组件都可以看作组件更新。普通的组件更新时，原有正在运行的组件将退出，更新完毕后重新运行更新后的组件，IMA 度量方法就能对这类更新进行加载时度量。组件动态更新(热更新)则是正在运行的组件不退出，运行时动态改变度量变量的状态，组件打补丁、安装插件、在线升级等是常见的组件动态更新方式。

组件动态更新时，TPM/TCM 已经完成了组件状态度量。然而 TPM/TCM 作为一个防篡改硬件，不提供 TPM/TCM 度量状态回滚(Roll Back)机制，无法重新度量反映更新后的状态，目前 TCG 的安全框架也没有提出动态更新的相应解决方法。

可信计算环境主要确保运行环境中硬件、固件和软件的真实性，即组件的真实性。要度量某个组件的完整性，只需要以组件 id 请求 MA，MA 对内存中运行的组件进程、依赖的内核模块和依赖的动态链接库进行度量，即对组件的全体度量变量进行度量，MA 度量输出组件度量值 χ 和度量日志 log。

对于某个标识为 id 的组件 c 而言，在组件发布时就确定好它的全部度量变量，设 $\mathrm{MC}=\{mc_1,\cdots,mc_k\}$是它的度量变量集合。则组件度量算法描述如下。

算法　$M(id)$。

输入　$\mathrm{MC}=\{mc_1,\cdots,mc_k\}$。

输出　如果执行成功，返回 log,χ；否则，返回错误代码。

具体执行过程如下。

(1) $\log:=\{\}$。

(2) MA 查找组件进程是否运行，若没有运行，则返回错误。

(3) MA 度量组件 c

① $\chi:=0^{160}$；

② For $j=1$ to k do

ⅰ. MA 检查度量变量 mc_j，计算度量值 $\omega_j:=H(mc_j)$。

ⅱ. 添加度量日志，$log:=log\cup\{(desc_j,\omega_j)\}$，$desc_j$ 为 mc_j 的具体描述信息。

ⅲ. 更新组件度量值，$\chi:=\chi\cdot\omega_j$。

③ return χ。

(4) return log。

由组件度量算法可以知道，组件 c 的度量结果为 $\chi:=0^{160}\cdot\omega_1\cdot\omega_2\cdot\cdots\cdot\omega_k$，而将上述算法简记为 $M(\mathrm{id}):=\{\log,\chi\}$。

2. 会话组件树计算

一般来说，TPM/TCM 芯片内部包含 24 个不可篡改的平台配置寄存器 PCR，PCR 只能通过扩展操作更新寄存器值。我们的原型系统中，新启动一个 Compartment 时，TCMG 从 PCR16～23 中选择一个物理 PCR 与之关联，用来存储 Compartment 的组件度量结果。当 RAI[a]实例进行远程证明时，将扩展 Compartment 的物理 PCR，由于多个 RAI 共享同一物理 PCR，好像为每个 RAI 虚拟一个 PCR 一样，因此称实例 RAI[a]扩展后的 PCR 为该 RAI 实例的 VPCR(Virtual PCR)，VPCR 值为 VPCR[a]。该方案中最多能够同时运行 8 个 Compartment，具体的物理 PCR 分配见表 16.1。

表 16.1　TPM/TCM 的物理 PCR 分配表

PCR 索引	PCR 用法
0～7	CRTM、BIOS、Motherboard、BootLoader 等度量结果
8	Hypervisor 层度量结果
9	可信服务层(MA、CM、TSMG 等)度量结果
10～15	系统运行时度量结果
16～23	Compartment 组件度量结果

RAI 要求平台满足一定的安全属性，亦即要求与该会话相关的组件满足一定的安全属性。将这些组件构建成一棵二叉杂凑树来表示 RAI 关联的计算环境，这棵树称之为会话组件树。一个新的 RAI 会话发起时，通信双方就交换安全策略协商好要证明的 RAI 组件，然后 TSMG 创建 RAI 会话组件树，TCMG 新建 VPCR 用来保存会话证明结果。当组件配置和状态发生变化时，会话组件树能够快速更新 RAI 组件状态，实现更新证明。

会话组件树是以组件为叶节点的杂凑树(Merkle Hash Tree，MHT)，在二叉杂凑树中(图 16.5)，叶节点是 RAI 关联的组件，包含组件的无碰撞杂凑值 χ，安全属性值 $p\in P$，叶节点的 χ 值按照前面的组件度量算法 $M(\mathrm{id}):=\{\log,\chi\}$计算得到。TSMG 根据组件属性证书对 log 中 MC 度量值进行验证，验证通过赋予组件安全属性 p，否则安全属性 $p=0^{160}$。RAI 组件对象构成了树 $\mathrm{MHT}(id_1,\cdots,id_m)$，树节点 n 存储的键值为 $k[n]=(\chi,p)$，节点 n 的标识 $l[n]$递归定义为：$l[n]=k[n]$，其中 $k[n]$是节点 $v[n]=(h_L,k[n],h_R)$的键值，$k[n]=H(l_L\parallel l_R)$。

h_L 和 h_R 分别是节点 n 的左子女节点和右子女节点，节点 n 的键值是左、右子树节点值连接的杂凑值，组件树的根节点 RAI 所有组件对象的无碰撞杂凑值。计算出组件会话树根节点值，然后扩展 TPM/TCM 相应的 PCR。会话组件树的节点、树根保存在 TSMG 运行

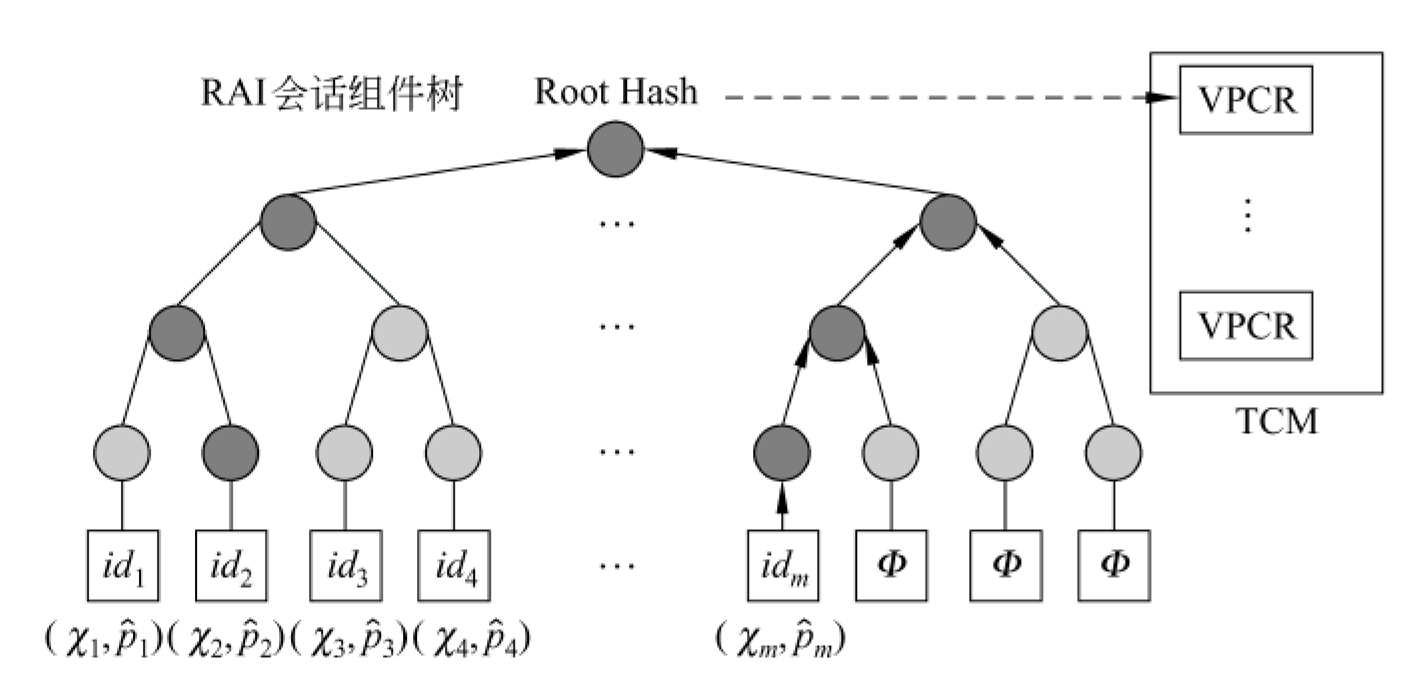


图 16.5　RAI 会话组件树

空间中，且只能够被 TSMG 操作。如果 RAI 组件个数为 $m=2^l$，那么会话组件树是一个高度为 $l+1$ 的满二叉树，若 $2^{l-1}<m<2^l$，会话组件树添加 2^l-m 个空节点 Φ 构成满二叉树。

RAI 会话实例的终端安全属性和配置状态用会话组件树描述，只要系统中 RAI 组件的任何度量变量发生改变，都将会反映到这棵会话组件树上来。TCMG 再用根节点值扩展相应的 TPM/TCM 的物理 PCR，那么 TPM/TCM 就能够标识各个 Compartment 的多个 RAI 会话状态。会话组件树很好地解决了会话相关组件的状态表示，并且便于组件动态更新状态标识。会话组件树的组件更新仅仅需要更新组件节点到 Root Hash 路径上全部节点值，就能反映出更新后的状态。

以组件为叶节点的 RAI 会话组件树计算简记为 $H(\mathrm{MHT}(id_1,\cdots,id_m))$，这里提供两种会话组件树的计算方法，一是使用度量值 χ 计算会话组件树，另一种是使用组件属性值 $p\in P$ 计算组件会话树。前者可用于直接二进制的远程证明，用应用程序的二进制杂凑值来表示可信计算环境，这种方法简单易行，但存在冗余证明和平台配置隐私泄露的缺陷。后者直接基于属性进行远程证明，证明灵活易于扩展，但需要颁发属性证书和验证属性是否撤销。

$$\begin{aligned}\mathrm{root}=&H(\mathrm{MHT}(id_1,\cdots,id_m))\Rightarrow\\&(H(\mathrm{MHT}(\chi_1,\cdots,\chi_m)),H(\mathrm{MHT}(p_1,\cdots,p_m)))\end{aligned}\tag{16-3}$$

如果会话组件树 $\mathrm{MHT}(id_1,\cdots,id_m)$ 有 $m(2^{l-1}<m\leqslant 2^l)$ 个叶节点，那么会话组件树的高度为 $h=l+1$，有 2^l-m 个空节点 Φ 构成满二叉树，空节点的度量值为 ϕ，定义空节点的计算规则如下。

(1) $\phi\cdot\phi=\phi$。

(2) $\phi\cdot\chi=\chi\cdot\phi=\chi$。

组件会话树的度量值和属性值的计算方法为

$$\begin{aligned}H(\mathrm{MHT}(id_1,\cdots,id_m))&=H(\mathrm{MHT}(\chi_1,\cdots,\chi_m))\\&=H(\mathrm{MHT}(\chi_1,\cdots,\chi_m,\underbrace{\phi,\cdots,\phi}_{2^l-m}))\\&=f(\chi_1,\cdots,\chi_m,\underbrace{\phi,\cdots,\phi}_{2^l-m};2^l)\end{aligned}\tag{16-4}$$

其中 $f(x_1,\cdots,x_{2^l};2^l)$ 为一递归计算函数，其函数定义为

$$f(x_1,\cdots,x_{2^t};2^t)=\begin{cases}f(x_1,\cdots,x_{2^{t-1}};2^{t-1})\cdot f(x_{2^{t-1}+1},\cdots,x_{2^t};2^{t-1}), & t>1\\ x_1\cdot x_2, & t=1\end{cases} \tag{16-5}$$

由式(16-3)、式(16-4)和式(16-5)可以计算得到根节点的度量值，同样会话组件树根节点属性值可以由式(16-3)、式(16-4)和式(16-5)计算得到：$H(\text{MHT}(id_1,\cdots,id_m))=f(p_1,\cdots,p_m,\underbrace{\phi,\cdots,\phi}_{2^t-m};2^t)$，计算结果最终更新相应的 VPCR，VPCR＝PCR・root，组件度量值和属性值验证也按照相同的方法进行。RAI 某个组件发生改变，不用重新计算整个会话组件树，只需要更新该组件节点到根节点路径上的节点值，亦即更新序列 s 所描述的从节点 $v=(h_L,k,h_R)$ 到根节点路径上的全部节点值。

$$s=(h_L,k,h_R;k_1,h_1;k_2,h_2;\cdots;k_r,h_r) \tag{16-6}$$

除了使用普通的杂凑树构建会话组件树外，可以借鉴虚拟单调计数器认证查找树[23]的构造方法，采用认证查找树(Authenticated Search Tree，AST)[24]构建 RAI 会话组件树 $\text{AST}(id_1,\cdots,id_m)$。以组件 ID 为关键字(或称键值)建立认证查找树，认证查找树的每个节点 n 都有唯一一个查找键值 $k[n]=(\text{id},\chi,p)$，如果 n_L 是节点 n 的左子女节点，则 $k[n_L]<k[n]$，如果 n_R 是节点 n 的右子女节点，则 $k[n]<k[n_R]$。$\text{AST}(id_1,\cdots,id_m)$树的节点 n 的标识 $l[n]$递归定义为：$l[n]=H(v[n])$，$v[n]=(h_L,k[n],h_R)$。如果节点 n 的左子女节点不存在，则 $l_L=\text{nil}$，右子女节点也一样。$\text{AST}(id_1,\cdots,id_m)$树节点 $v=(h_L,k,h_R)$到根结点 v_r 按照序列(16-6)进行扩展计算。令 $l_0=H(v)$，对于 $0<j\leqslant r$，认证查找树节点可以进行以下递归计算，即

$$l_j=H(v_j),\quad v_j=\begin{cases}(l_{j-1},k_j,h_j), & k<k_j\\ (h_j,k_j,l_{j-1}), & k_j<k\end{cases} \tag{16-7}$$

认证查找树是以组件 ID 为节点键值构建，而杂凑二叉树是以组件度量值(或属性值)为叶节点构建，两种组件会话树方式都可以描述 RAI 会话相关组件的运行状态。RAI 证明会话开始建立时，双方交换证明策略，其中包含了 RAI 要证明的组件 ID(或安全属性)，TSMG 根据组件 ID 建立组件会话树，验证时，Requester 则通过证明策略中的组件 ID 构建组件会话树进行组件运行状态和属性的验证。

3. Multi-RAI 证明

Compartment 启动时，TCMG 就分配一个物理 PCR 与之关联，TCMG 必须能够完成 Multi-RAI 的并行证明，在未使用隔离技术的单个系统(如 Windows 或 Linux 系统)，多个实例并发证明的问题将更为突出。Multi-RAI 证明除了满足并发性外，还必须满足证明的一致性和防重放攻击。

RAI 组件状态最终以组件树的形式聚集为 VPCR 值，设实例 RAI[a]的组件状态聚集到组件树根节点值为 θ，TCMG 扩展相应的物理 PCR，得到 RAI[a]的 VPCR 值，VPCR[a]＝PCR・θ。组件度量日志、RAI 会话组件树、根节点值 θ 和 VPCR 值共同构成了证明状态数据 D。TCMG 进行物理 PCR 扩展后，会相应地增加单调计数器的值，用于保证证明状态的一致性和防止恶意攻击者和程序的重放攻击。

RAI 实例建立时，TCMG 会为 RAI 生成证明凭证，以后每一次 RAI 组件状态更新，

TCMG 会生成相应的更新凭证。证明凭证和更新凭证都由 TCMG 使用算法 $A(i,a,D,t)$ 生成,凭证格式为:[θ_t,PCR,n,t,Sign(sk,$H(n \| t \|$ PCR))],证明凭证和更新凭证的区别是在于随机数 n,证明凭证中 n 是 Requester 发送的 Nonce,而更新凭证为 TPM/TCM 内部生成的随机数。Compartment 的多个 RAI 并发的扩展同一物理 PCR,这样就构成一条与单调计数器关联的更新凭证链(图 16.6),对于 RAI 实例 a 建立初始会话时,只需要提供时刻 t_0 时的证明凭证和它之前 t_0-1 的更新凭证(凭证 RAI_ID≠a),就能证明 RAI 实例运行环境的可信。

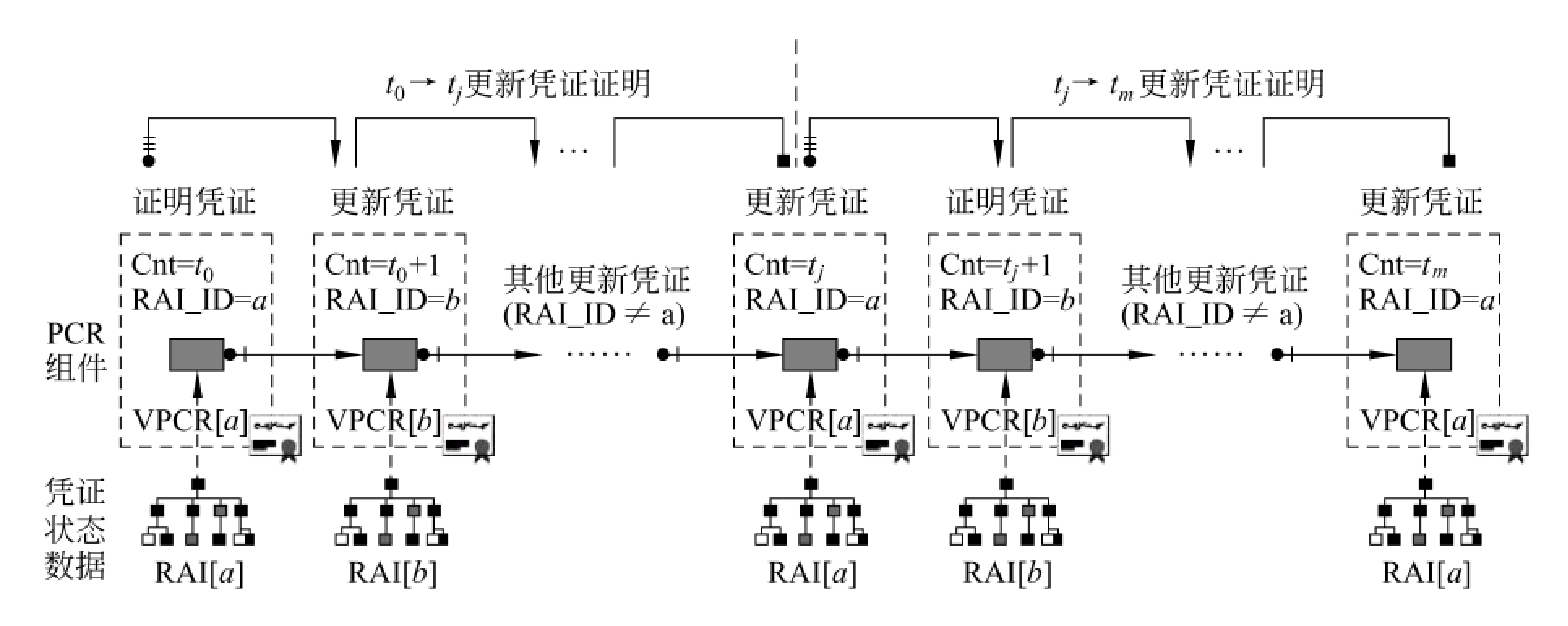


图 16.6　Multi-RAI 更新证明证据

当需要证明 t_m 时刻组件更新后的状态可信,TSMG 只需查找最近一次 RAI[a]凭证 u_j,提供更新凭证证据 $f=(u_j,u_{j+1},\cdots,u_m)=U(i,a,D_j,D_m)$。验证时 Requester 首先重构时刻 t_m 的证明状态数据,主要是根据证明策略请求的组件 ID 建立会话组件树,验证根节点 θ_m 可信,然后验证更新凭证链有效。这种方法有效地防止了 RAI[a]重放攻击,因为攻击者重放 t_k 时刻的更新证明,攻击者必须满足以下两个条件才能成功:①伪造时刻 $t_k(t_k=t_m)$ 的更新凭证;②伪造一条从 $t_j \rightarrow t_m$ 的更新凭证链。这就要求攻击者能够破解无碰撞杂凑函数和 TPM/TCM 的签名。

RAI 证明与 TLS/SSL 协议的认证方式相同,也分为单向证明和双向证明,双向证明不过是 RAI 在 Requester 一方的重用,这里主要探讨 RAI 单向证明。RAI 证明时,首先证明系统 TCB 的可信,采用二进制证明就能满足要求,IMA 度量体系[4]和 TLC 原型系统[1]实现了这种方法。该方案中,证明 TCB 启动时各个部件及 PCR0～PCR9 的安全性。然后再证明与 RAI 直接关联的用组件表示的可信计算环境的安全性,证明时 TCMG 执行 $A(i,a,D,t)$算法为 RAI 实例生成证明凭证,TSMG 借助证明凭证和证明状态数据 D 进行证明。由于操作系统、应用软件自身缺陷和漏洞的暴露,以及攻击技术的提高,原有的组件配置将不再满足某一安全属性,原有颁发的属性证书将无效,证书发布权威机构将予以撤销。一般来说,验证组件属性证书是否被撤销有两种方法:一种方法是 TCB 本地验证,CM 验证管理的属性证书是否有效,如果运行的组件的属性证书已经被撤销,立即通知系统管理员,通知 Requester 某组件运行状态不安全;另一种方法是远程验证,Requester 以(id,p)查询证书权威机构,验证属性证书是否被撤销。由于存储在本地用于验证的 CRL 存在被替换和版本回滚(Version-Rollback)的危险,文献[9]提供了一种 TPM 封装存储 CRL,定期更新

CRL 的本地验证属性证书的方法。为了降低本地系统的复杂性，采用远程验证组件属性的方法，其缺陷是在仅进行证明组件属性的 RAI 证明中，降低了平台组件的隐私性。

前面对 RAI 证明的相关问题及解决方法进行了概述，下面具体阐述 RAI 证明流程，为简化起见将忽略对 TCB 状态的证明。

(1) Requester 向 RAI 发送 nonce 和 attestpolicy。

(2) RAI 解析 attestpolicy，并完成以下操作。

① MA 度量组件：$M(id) := \{\log, \chi\}$。

② TSMG 构建 RAI 会话组件树，形成证明状态数据 D。

③ TCMG 请求 TPM/TCM 生成证明凭证，$s \leftarrow A(i,a,D_j,t_j)$，PCR 初值为 t_j-1 的 VPCR 值，记为 PCR_0，其中

$$s = [\theta_j, \mathrm{VPCR}_j, \mathrm{nonce}, t_j, \mathrm{Sign}(\mathrm{sk}, H(\mathrm{nonce} \parallel t_j \parallel \mathrm{VPCR}_j))] \tag{16-8}$$

(3) RAI 向 Requester 发送 s，log 和 PCR_0。

(4) Requester 完成以下验证操作。

① Requester 验证证明凭证，$(\mathrm{VPCR}_j, t_j) \leftarrow A_V(s,i,a,\mathrm{pk})$。

② 验证组件属性是否被撤销，然后重构会话组件树，计算根节点 θ'_j，验证 $\theta_j \overset{?}{=} \theta'_j$。

③ 验证 $\mathrm{VPCR}_j \overset{?}{=} \mathrm{PCR}_0 \cdot \theta'_j$。

④ 前两步都验证成功，Requester 对 t_j 和 VPCR_j 颁发确认凭证。

RAI 实例收到 Requester 的确认凭证后，TSMG 会将 t_j 和 VPCR_j 标识的证明凭证作为最近的一次更新凭证，也就是说，这个证明凭证所描述的状态是 RAI 实例的计算环境的最新状态，这样以便以后的更新证明。

Multi-RAI 证明建立安全会话后，随着正常的软件更新、升级，或者恶意病毒和黑客的攻击，导致会话关联组件的配置和运行状态发生改变，此时原有建立的信任关系将被破坏，因此需要重新证明。RAI 更新证明流程如下。

(1) RAI 更新证明状态数据并产生更新凭证证据。

① TSMG 更新实例 RAI[a]的最近一次证明状态数据，$D_j \rightarrow D_m$，接着计算会话组件树根节点 θ_m。

② TCMG 扩展 PCR，$\mathrm{VPCR}_m = \mathrm{VPCR}_{m-1} \cdot \theta_m$，然后生成更新凭证 $u_m \leftarrow A(i,a,D_m,t_m)$，其中 $u_m = [\theta_t, \mathrm{VPCR}_m, n, t_m, \mathrm{Sign}(\mathrm{sk}, H(n \parallel t_m \parallel \mathrm{VPCR}_m))]$。

③ TSMG 生成更新凭证证据 $f \leftarrow U(i,a,D_j,D_m)$。

$$f = (u_j, u_{j+1}, \cdots, u_m)$$

(2) RAI 向 Requester 发送 f 和 log。

(3) Requester 完成以下验证操作。

① Requester 验证更新凭证 u_m，$(\mathrm{VPCR}_m, t_m) \leftarrow A_V(u_m,i,a,\mathrm{pk})$。

② 验证更新的组件属性是否被撤销，计算出会话组件树根节点 θ'_m，验证 $\theta_m \overset{?}{=} \theta'_m$ 和 $\mathrm{VPCR}_m \overset{?}{=} \mathrm{VPCR}_{m-1} \cdot \theta'_m$。

③ 验证更新凭证证据，$\{(\mathrm{VPCR}_j, t_j), (\mathrm{VPCR}_m, t_m)\} \leftarrow U_V(f,i,a,\mathrm{pk})$。

④ 如果都验证成功，Requester 对 t_m 和 VPCR_m 颁发确认凭证。

(4) RAI 收到 Requester 的确认凭证后，将凭证 u_m 作为最近的一次更新凭证。

上述过程中如果 RAI 更新证明验证失败，Requester 将切断 RAI 的 SSL 网络连接。如果更新证明仅仅是证明属性，RAI 实例的会话组件尽管配置发生改变，但安全属性没有发生改变，此时 RAI 没有必要进行更新证明。

16.1.3 远程证明方案的安全性和效率分析

在 H 是无碰撞杂凑函数、Sign 是安全的数字签名方案的前提下，下面简要说明可信计算环境证明方案 $\Gamma=(G,M,A,A_V,U,U_V)$ 是安全的。

首先，令 u 和 u' 都是更新凭证，满足 $t\leftarrow A_V(u,i,a,\mathrm{pk})$ 和 $t\leftarrow A_V(u',i,a',\mathrm{pk})$，且 $a\neq a'$，由于 TPM/TCM 签名方案假定是安全的，Compartment i 在时刻 t 只能返回一个签名，因此这两个凭证 u 和 u' 是完全相同的，凭证包含时刻 t 的 PCR 值 VPCR_t。对于 Compartment i 而言 $t-1$ 存在唯一更新凭证 u_{t-1}，凭证包含 VPCR_{t-1}。$\mathrm{VPCR}_t=\mathrm{VPCR}_{t-1}\cdot\theta_t$，所以更新凭证 u 和 u' 关联的会话组件树的根节点 θ_t 相等，唯一的可能便是 RAI 实例 a' 可以破解无碰撞杂凑函数 H。

其次，令 f 是更新凭证证据，u 为更新凭证，满足 $\{(\mathrm{VPCR}_j,t_j),(\mathrm{VPCR}_m,t_m)\}\leftarrow U_V(f,i,a,\mathrm{pk})$，$t_h\leftarrow A_V(u,i,a,\mathrm{pk})$，$t_j<t_h\leqslant t_m$。$f=(u_j,\cdots,u_m)$ 包含了时刻 t_m 的 VPCR_m 不可伪造签名凭证 u_m，更新 Oracle 将 PCR 值由 VPCR_{m-1} 扩展为 VPCR_m，$\mathrm{VPCR}_m=\mathrm{VPCR}_{m-1}\cdot\theta_m$。重复上述过程，$f$ 包含时刻 t_h 的 VPCR_h 不可伪造签名凭证 u_h，$\mathrm{VPCR}_m=\mathrm{VPCR}_h\cdot\theta_{h+1}\cdot\cdots\cdot\theta_m$，由于 H 为无碰撞杂凑函数，在时刻 t_h，TPM/TCM 仅仅只能返回唯一的更新凭证，所以凭证 u_h 和 u 必定存在一个是伪造的。

由上述讨论可知，在 H 是无碰撞杂凑函数、Sign 是安全的数字签名方案下，Multi-RAI 证明和更新证明是安全的。

假定 Compartment 中共有 N 个组件，同时有 I 个 RAI 会话实例运行于 Compartment 中，在最坏的情况下，每个 RAI 实例平均拥有 N/I 个 RAI 会话组件。RAI 会话树的每个节点计算、PCR 扩展等，TPM/TCM 都必须作为一个原子操作，不允许被中断。这样对于实例 RAI[a]的一个更新凭证证据而言，TPM/TCM 要执行 $O(I(1+\log(N/I)))$ 个原子操作。

定义 ω 为实例 RAI[a]在一个时间单位内发生组件更新，执行更新证明的概率，为简化起见，假定每个组件更新概率都为 ω。γ 为 TPM/TCM 执行一个原子操作指令的平均计算时间。J 为一个 RAI 组件时间在同一个时间单位上请求更新的组件个数期望值($0\leqslant J\leqslant N/I$)，$T$ 为同一个 RAI 实例连续两次请求更新证明的时间间隔期望值。

对于 Multi-RAI 证明，更新证据计算过程 $f\leftarrow U(i,a,D_j,D_m)$ 是 $[t_j,t_m]$ 时间段内计算量最大、最耗时的步骤。产生更新证明证据的每个更新凭证 $u_k(j\leqslant k\leqslant m)$ 计算上要耗费 $1+J+\log(N/I)$ 个原子操作，其中 J 为组件 Hash 值更新计算原子操作时间，1 为更新物理 PCR 的原子操作时间。

$$T=\gamma I(1+J+\log(N/I)) \tag{16-9}$$

同一 Compartment 中，RAI 实例在时间 T 内组件不更新，不执行更新证明的概率为 $(1-\omega)^T$，所以RAI 实例在同一时间有组件更新的概率为 $1-(1-\omega)^T$，则平均有 $\frac{N}{I}(1-(1-\omega)^T)$ 个组件同时更新，因此

$$J=\frac{N}{I}(1-(1-\omega)^T)\approx\omega NT/I \tag{16-10}$$

同一个 Compartment 有多个 RAI 同时请求更新证明，即同时有多个 RAI 的会话关联组件发生更新，同时更新时，TCMG 将按顺序依次更新 PCR 进行证明，在同一单位时间内，TCMG 只能生成一个 RAI 实例的更新凭证，其他的 RAI 将进入等待队列。若 X 为执行更新证明概率为 ω 的时间随机变量，X 服从几何分布，$E(X)=1/\omega$。那么平均 $1/\omega$ 个单位时间，RAI 执行一次更新协议，平均 $(1-\omega)/\omega$ 个单位时间没有 RAI 执行更新协议。因此，要求同一个 RAI 两次更新请求间隔时间期望满足

$$T \leqslant (1-\omega)/\omega \tag{16-11}$$

将式(16-10)代入式(16-9)得到，$T=\gamma I+\omega\gamma NT+\gamma I\log(N/I)$，因此

$$T = \frac{\gamma I(1+\log(N/I))}{1-\omega\gamma N} \tag{16-12}$$

然后对公式(16-12)进行求导，计算 T 的最大值，$\frac{\mathrm{d}T}{\mathrm{d}I}=\frac{\gamma\log(N/I)}{1-\omega\gamma N}$，因此 I 满足：$\frac{\gamma\log(N/I)}{1-\omega\gamma N}=0$，$I=N$。代入式(16-12)中得到

$$T = \frac{\gamma N}{1-\omega\gamma N} \tag{16-13}$$

同一 RAI 实例更新证明的平均间隔时间与组件更新的概率、TPM/TCM 的原子操作处理时间和 Compartment 组件的个数密切相关，等式(16-13)给出了它们之间的相互关系。TPM/TCM 原子操作受 TPM/TCM 物理芯片和硬件 I/O 效率影响，系统的组件更新概率是一个不可预知的因素，可以通过分隔 Compartment 的组件，使系统满足更新证明效率要求。将式(16-13)代入不等式(16-11)进行化简得到

$$N \leqslant \frac{1-\omega}{\omega\gamma(2-\omega)} \tag{16-14}$$

在系统组件更新概率和 TPM/TCM 原子操作速度固定的前提下，可以按照不等式(16-14)限制Compartment 中运行的组件个数，以避免同一 Compartment 中 Multi-RAI 并发更新证明，而影响可信服务层(包括 TSMG、TCMG、CM 等)的处理效率。

我们对 Multi-RAI 证明方案的系统实现问题也进行了讨论，具体讨论可参阅文献[25]。

16.2　基于 TCM 的属性证明协议

本节介绍一种新型的、基于双线性映射的属性证明协议(简称为 PBA-BM 协议)。该协议建立在实用的完整性管理和在线可信第三方基础上，提供一个基于 TCM 安全芯片的属性证明协议；该协议基于配置承诺和双线性对 CL-LRSW 签名，实现了远程属性证明的真实性和配置隐私保护等安全属性，并且该协议在随机 Oracle 模型下是可证明安全的。与原有的 PBA 协议相比，PBA-BM 简化了属性证明的属性撤销验证流程，具有更短的签名长度和更少的计算量。

16.2.1　属性证明模型

基于属性的远程证明减少了繁琐的计算配置的证明和验证，由 TCM 安全芯片直接向远程验证方证明平台的配置状态所达到的安全属性，满足验证方的安全需求。在属性证明

模型中，主要的协议参与者包括证明方 Π(包含主机 H 和安全芯片 M)，验证方ς和属性权威机构 T。证明方证明平台的状态与属性凭证所描述的安全属性一致，在证明方平台中假定安全芯片 M 是真实不可篡改的；验证方验证证明数据是否满足安全属性要求；属性权威机构 T 作为一个在线的可信第三方，负责颁发平台的安全属性证书，以及验证属性证书是否撤销。在属性证明参与者中，TCM 安全芯片和可信第三方都是完全可信的，属性证明中最常遇到的两类攻击便是伪造欺骗攻击和获取平台配置隐私攻击，伪造欺骗攻击是指被破解的主机 H 通过截获、伪造证明消息，使得即使平台处于一种不可信状态，仍然能够通过远程验证方ς的验证；获取平台配置隐私攻击则是指恶意的远程验证方ς通过分析证明方提供的证明数据，获取相应的平台配置信息以便实施针对配置漏洞展开攻击。

下面从属性证明流程来阐述属性证明模型，参见图 16.7。

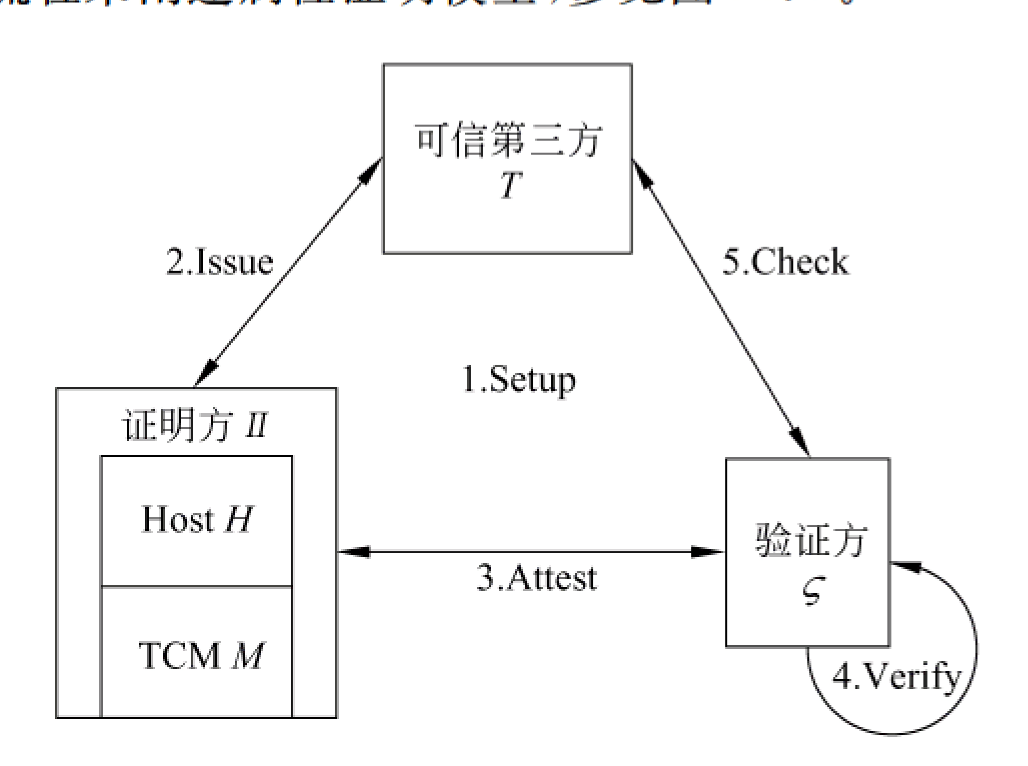


图 16.7 属性证明流程图

属性证明模型中进行属性远程证明包含的关键操作步骤如下。

(1) Setup。建立属性证明系统的系统参数，使用密钥生成算法 $G(1^k)\to(\mathrm{sk},\mathrm{vk})$生成参与实体 M 和 T 的密钥对。为 TCM 芯片生成公、私钥 Msk、Mvk；为 T 生成公、私钥 sk、vk。

(2) Issue。Π 请求 T 为当前配置 cs 生成属性凭证，T 评估平台的配置 cs 所达到的安全属性(设定为 ps)，T 使用私钥 sk 对配置属性对(cs，ps)进行签名，即颁发属性证书 cre，然后将属性证书发送回 Π，Π 保存配置属性证书 cre 以便随后进行证明。

(3) Attest。验证方ς以随机数 N_v 挑战请求证明者 Π 证明其当前配置状态 cs 满足某一属性 ps，安全芯片 M 首先计算 cs 的承诺 C，然后对配置承诺 C 进行签名，H 根据配置承诺 C 和配置属性证书 cre 进行知识签名，Π 最后输出基于属性证明的签名 σ_{PBA}，将 σ_{PBA} 发送给ς进行验证。

(4) Verify。验证方ς接收到 Π 的证明数据后，首先验证请求的随机数 N_v 是否一致，紧接着验证安全芯片的签名 δ，然后验证关于承诺 C 和属性证书 cre 的知识签名，最后验证属性 ps 是否已经撤销。如果上述过程都通过验证，则 Π 证明成功，否则失败。

对于 ς 验证属性是否撤销的方法多种多样，PBA 利用的是零知识证明 $\mathrm{cs}\notin \mathrm{CS}_{\mathrm{revoked}}$的方法；PBA-RS 则是双方协商证明的配置集合 CS，保证了需要证明的配置一定不会是撤销的；我们的方案则是通过在线的可信第三方进行属性撤销验证(即 Check 过程)。

(5) Check。ς向 T 发送属性验证数据，T 使用私钥 sk 获得证明过程中使用的属性证书

cre，然后查询属性数据库，判断该配置属性(cs，ps)是否已经被撤销，最后 T 将验证结果以安全的方式传递给ς。

要在远程证明过程中保证 Π、ς双方利益的前提下，同时实现 Π 平台的安全属性满足ς的安全要求，则属性证明模型必须满足以下两方面的安全需求。

(1) 不可伪造性(Unforgeability)。令 $\mathrm{Game}_A^{\mathrm{att\text{-}fg}}(1^k)$是参与方 Π、ς、T、A 之间的伪造证明的攻击交互游戏过程，A 选择了一个有效的安全芯片 M，向ς证明配置属性(cs′，ps′)，其中(cs′，ps′)$\notin$CS，CS 是 T 可以接受的可信配置属性集合。设定 A 能够截获、修改、转发任何参与者的通信消息，即当诚实的参与方按照协议正常执行，A 通过发送消息 send(E,m)和查询 Oracle O 攻击属性证明系统，其中 $E\in\{H,M,V\}$。最终 A 输出了能够被ς接受的未查询过 O 的 PBA 签名 σ_{PBA}，则 A 赢得胜利。设 A 赢得 $\mathrm{Game}_A^{\mathrm{att\text{-}fg}}(1^k)$的优势概率为 $\mathrm{Succ}_A^{\mathrm{att\text{-}fg}}(1^k)=\Pr[\mathrm{Game}_A^{\mathrm{att\text{-}fg}}(1^k)=\mathrm{win}]$，如果 $\mathrm{Succ}_A^{\mathrm{att\text{-}fg}}(1^k)$关于 k 是可忽略的，则属性证明是不可伪造的。

(2) 配置隐私保护(Configuration Privacy)。令 $\mathrm{Game}_A^{\mathrm{att\text{-}fg}}(1^k)$是参与方 Π、ς、T、A 之间窃取配置隐私的攻击交互游戏过程，对于同一安全属性对应的 n 个可能的配置对(cs$_i$，ps)，设定 A 能够以优势概率 $\mathrm{Adv}[A_{\mathrm{PBA}}^{\mathrm{cf\text{-}prv}}]=|\Pr[b'=j]-1/n|$攻破属性证明的配置隐私。如果对于任何多项式时间的攻击敌手 A，攻击配置隐私的优势概率 $\mathrm{Adv}[A_{\mathrm{PBA}}^{\mathrm{cf\text{-}prv}}]$是可忽略的，则就说属性证明方案满足配置隐私保护属性。

16.2.2 知识签名和 CL-LRSW 签名方案

知识签名允许一方在不泄露任何有用信息的情况下证明他知道一个秘密值，这种工具本质上是知识的零知识证明或最小泄露证明。

基于离散对数的零知识证明协议已有很多相关研究成果，为了描述这类协议，采用文献[26]中的标记法来描述基于离散对数的零知识证明协议，例如

$$\mathrm{PK}\{\alpha,\beta,\delta: y = g^\alpha h^\beta \wedge \tilde{y} = \tilde{g}^\alpha \tilde{h}^\delta \wedge (u \leqslant \alpha \leqslant v)\}$$

表示“关于整数 α、β、δ 的零知识证明，并且 $y=g^\alpha h^\beta$，$\tilde{y}=\tilde{g}^\alpha\tilde{h}^\delta$，$u\leqslant\alpha\leqslant v$ 同时成立”。其中，y、g、h、$\tilde{y}$、$\tilde{g}$、$\tilde{h}$ 分别是群 $G=\langle g\rangle=\langle h\rangle$和群 $\tilde{G}=\langle\tilde{g}\rangle=\langle\tilde{h}\rangle$中的元素。可利用 Fiat-Shamir 启发式算法[27]将零知识证明协议转化为对消息 m 的知识签名[28]，可记作：$\mathrm{SPK}\{(\alpha):y=g^\alpha\}(m)$。

CL-LRSW 签名方案的安全性是建立在 LRSW 假设基础之上，因此，这里先介绍一下 LRSW 假设。LRSW 假设是指：令 $G=\langle g\rangle$是循环群，$X,Y\in G$，$X=g^x$，$Y=g^y$。假定有一 Oracle O，当输入 $m\in Z_q$，随机选择 $a\in G$，输出 3 元组(a,a^y,a^{x+mxy})，则一定不存在有效的攻击敌手通过多项式次数内查询 Oracle O，输出了未查询过的消息 m 对应的元组(m,a,b,c)，满足 $m\neq0$，$b=a^y$，$c=a^{x+mxy}$。

我们的属性证明协议建立在 CL-LRSW 签名方案的基础上，与普通签名方案不同的是，CL-LRSW 签名方案使用了双线性映射 e，而且能够快速、有效地进行知识签名构建密码协议。CL-LRSW 签名方案包含以下几个算法。

(1) Key generation。用于生成签名方案所需要的密钥对，执行 Setup 算法生成$(q,G,G_{\mathrm{T}},g,g_{\mathrm{T}},e)$，随机选择 $x\in_R Z_q$ 和 $y\in_R Z_q$，计算 $X=g^x$，$Y=g^y$，设置 sk$=(x,y)$，pk$=$

(q,G,G_T,g,g_T,e,X,Y)。

(2) Signature。输入消息 m，私钥 $sk=(x,y)$ 和公钥 $pk=(q,G,G_T,g,g_T,e,X,Y)$，随机选择 $a\in_R G$，输出签名 $\sigma=(a,a^y,a^{x+mxy})$。

(3) Verification。输入公钥 $pk=(q,G,G_T,g,g_T,e,X,Y)$，消息 m 和待验证的签名 $\sigma=(a,b,c)$，然后检查 $e(a,Y)=e(g,b)$，$e(X,a)\cdot e(X,b)^m=e(g,c)$ 是否成立。

上面介绍了 CL-LRSW 签名方案只签名一个消息的简单情形，实际上 CL-LRSW 签名方案可以推广到签名多个消息的情形。我们的属性证明协议中就使用了签名两个消息的情形。下面介绍 CL-LRSW 签名方案签名两个消息的情形。

(1) Key generation。密钥生成算法用于生成签名方案所需要的密钥对，执行 Setup 算法生成 (q,G,G_T,g,g_T,e)，随机选择 $x\in_R Z_q$，$y\in_R Z_q$ 和 $z\in_R Z_q$，计算 $X=g^x$，$Y=g^y$，$Z=g^z$，设置 $sk=(x,y,z)$，$pk=(q,G,G_T,g,g_T,e,X,Y,Z)$。

(2) Signature。输入消息 (m,r)，私钥 $sk=(x,y,z)$ 和公钥 $pk=(q,G,G_T,g,g_T,e,X,Y,Z)$，随机选择 $a\in_R G$，计算 $A=a^z$，$b=a^y$，$B=A^y$，$c=a^{x+xym}A^{xyr}$，输出签名 $\sigma=(a,A,b,B,c)$。

(3) Verification。输入公钥 pk，消息 (m,r) 和待验证的签名 σ，然后检查 $e(a,Z)=e(g,A)$，$e(a,Y)=e(g,b)$，$e(A,Y)=e(g,B)$，$e(X,a)\cdot e(X,b)^m\cdot e(X,B)^r=e(g,c)$ 是否成立。

文献[29]中证明了 CL-LRSW 签名方案在 LRSW 假设下是安全的，即能够抵抗适应性选择消息攻击。我们的协议采用该方案对承诺消息进行签名，即对消息对 (m,r) 进行 CL-LRSW 签名。

16.2.3 基于双线性对的属性证明协议

1. 属性证明协议流程

(1) Setup

设置属性证明系统安全参数 l_q、l_H、l_ϕ，其中，l_q 为素数 q 阶群的阶的长度，l_H 为用于 Fiat-Shamir 启发式算法[27]的 Hash 函数的输出长度，l_ϕ 为用于控制零知识证明的安全参数的长度。令 H 是协议中用到的强无碰撞的 Hash 函数，$H:\{0,1\}^*\to\{0,1\}^{l_H}$。$T$ 选择两个阶为素数 q 的群 $G=\langle g\rangle$，$G_T=\langle g_T\rangle$，以及群 G 和 G_T 之间的双线性映射 e，$e:G\times G\to G_T$，然后 T 按照 CL-LRSW 签名的参数设置私钥 $sk=(x,y,z)$，公钥 $pk=(q,G,G_T,g,g_T,e,X,Y,Z)$。

(2) Issue

Π 所在平台进行平台配置状态完整性收集，然后以当前平台配置 cs 请求 T 颁发配置属性证书。T 根据配置状态信息进行安全属性评估，设定评估后的安全属性为 ps。T 对(cs, ps)颁发配置属性证书 cre，即使用私钥进行 CL-LRSW 签名，签名结果为 (a,A,b,B,c)，其中 $a\in_R G$，$A=a^z$，$b=a^y$，$B=A^y$，$c=a^{x+x\cdot y\cdot cs}A^{x\cdot y\cdot ps}$。令 $\sigma=(a,A,b,B,c)$，则平台配置属性证书 $cre=((cs,ps),\sigma)$。

T 将平台配置属性证书发送给 Π，Π 按照以下方法验证属性证书的有效性，Π 验证通过后保存属性证书，以便以后进行属性证明。

$$e(a,X)\overset{?}{=}e(g,A),\quad e(a,Y)\overset{?}{=}e(g,b),\quad e(A,Y)\overset{?}{=}e(g,B),$$
$$e(X,a)\cdot e(X,b)^{cs}\cdot e(X,B)^{ps}\overset{?}{=}e(g,c)$$

T 颁发的 CL-LRSW 签名具有随机化的属性，选择随机数 $r'\in Z_q^*$，计算 $a'=a^{r'}$，$b'=$

$b^{r'}$，$A'=A^{r'}$，$B'=B^{r'}$，$c'=c^{r'}$，则随机化后的 $\sigma'=(a',A',b',B',c')$ 也是元组(cs，ps)的 CL-LRSW 签名。

(3) Attest

ς以随机数 $N_v \in_R \{0,1\}^{l_H}$ 挑战请求 Π 所在平台进行属性证明，Π 接收到请求后，由主机平台 H 调用 TCM 芯片(M)进行配置证明。M 在内部使用随机数发生器 RNG 生成 $r_h, r_0 \in_R Z_q^*$，$N_t \in_R \{0,1\}^{l_\phi}$，然后计算 $h_T = g_T^{r_h} \in G_T$，利用 12.1 节介绍的 Pedersen 承诺方案[30]生成 cs 的承诺 $C = g_T^{cs} h_T^{r_0}$，使用 TCM 芯片的 PIK(Platform Identity Key)对承诺进行签名，签名为 $\delta = \text{Sig}_M(C, N_v \| N_t)$。

M 将 g_T、h_T、C、r_0、δ、N_t 发送给主机 H 进行属性证明签名。

H 随机化平台属性证书签名 σ，$a'=a^{r'}$，$b'=b^{r'}$，$A'=A^{r'}$，$B'=B^{r'}$，$c'=c^{r'r^{-1}}$，其中 r'，$r \in_R Z_q^*$。H 选择随机数 $t_1, t_2 \in_R Z_q^*$，计算 $\sigma_o = \sigma \cdot g^{t_1+t_2}$，然后计算以下参数：

$$v_x = e(X, a'),\quad v_{xy} = e(X, b'),\quad v_s = e(g, c'),\quad v_{xyz} = e(X, B')$$

在这一步可以优化协议计算，主机可以根据属性证书预先进行 Pairing 计算，$\bar{v}_x = e(X,a)$，$\bar{v}_{xy} = e(X,b)$，$\bar{v}_s = e(g,c)$，$\bar{v}_{xyz} = e(X,B)$，当 H 接收到证明挑战请求时，基于上述预先计算的值能非常容易地计算出 $v_x = (\bar{v}_x)^{r'}$，$v_{xy} = (\bar{v}_{xy})^{r'}$，$v_s = (\bar{v}_s)^{r'}$，$v_{xyz} = (\bar{v}_{xyz})^{r'}$。

紧接着由 H 按照以下步骤计算知识签名：

$$\text{SPK}\{(cs, r_0, r, t_1, t_2) \mid v_x v_{xy}^{cs} v_{xyz}^{ps} = v_s^r \wedge C = g_T^{cs} h_T^{r_0} \wedge d_1 = X^{t_1} \wedge d_2 = Y^{t_2}\}(N_v, N_t)$$

① H 选择随机数 $R_1, R_2, R_3, R_4, R_5 \in_R Z_q^*$，计算 $\bar{T}_1 = v_s^{R_3}(v_x v_{xy}^{R_1} v_{xyz}^{ps})^{-1}$，$\bar{T}_2 = g_T^{R_1} h_T^{R_2}$，$\bar{d}_1 = X^{R_4}$，$\bar{d}_2 = Y^{R_5}$。

② H 计算

$$c_H = H(q \| g \| X \| Y \| a' \| b' \| c' \| A' \| B' \| g_T \| h_T \| C \| \sigma_o \| d_1 \| d_2 \| v_x \| v_{xy} \| v_{xyz} \| v_s \| \bar{d}_1 \| \bar{d}_2 \| \bar{T}_1 \| \bar{T}_2 \| N_v \| N_t)$$

③ H 计算

$$s_1 = R_1 - c_H \cdot cs \bmod q,\quad s_2 = R_2 - c_H \cdot r_0 \bmod q,$$
$$s_3 = R_3 - c_H \cdot r \bmod q,\quad s_4 = R_4 - c_H \cdot t_1 \bmod q,$$
$$s_5 = R_5 - c_H \cdot t_2 \bmod q$$

最后 H 输出基于属性证明的签名 $\sigma_{PBA} = (\delta, C, a', b', c', A', B', c_H, s_1, s_2, s_3, s_4, s_5)$，将相应的证明数据传送给$\varsigma$。

(4) Verify

验证方ς知道可信第三方验证属性证书的公钥 $pk = (q, G, g, G_T, e, X, Y, Z)$，而且 Π 和ς之间需要证明的安全属性 ps 也是在通信过程中预先协商好。当ς接收到关于(N_v, N_t)的基于属性证明的签名 σ_{PBA}，ς执行以下步骤进行验证。

① ς验证 $h_T \overset{?}{\in} G_T$，使用 TCM 芯片的 PIK 公钥验证其对承诺的签名：$\text{Verf}_M(\delta, C, N_v \| N_t) \overset{?}{=} \text{true}$

② ς验证 $e(a', Z) \overset{?}{=} e(g, A')$，$e(a', Y) \overset{?}{=} e(g, b')$，$e(A', Y) \overset{?}{=} e(g, B')$

③ ς计算

$$\hat{v}_x = e(X, a'), \quad \hat{v}_{xy} = e(X, b'), \quad \hat{v}_s = e(g, c'), \quad \hat{v}_{xyz} = e(X, B');$$

$$\hat{T}_1 = v_s^{s_3} v_{xy}^{-s_1} (v_x v_{xyz}^{\mathrm{ps}})^{c_H - 1}, \quad \hat{T}_2 = g_T^{s_1} h_T^{s_2} C^{c_H}, \quad \hat{d}_1 = X^{s_4} d_1^{c_H}, \quad \hat{d}_2 = Y^{s_5} d_2^{c_H}$$

④ 紧接着ς验证

$$c_H \stackrel{?}{=} H(q \parallel g \parallel X \parallel Y \parallel a' \parallel b' \parallel c' \parallel A' \parallel B' \parallel g_T \parallel h_T \parallel C \parallel \sigma_o$$

$$\parallel d_1 \parallel d_2 \parallel v_x \parallel v_{xy} \parallel v_{xyz} \parallel v_s \parallel \hat{d}_1 \parallel \hat{d}_2 \parallel \hat{T}_1 \parallel \hat{T}_2 \parallel N_v \parallel N_t)$$

⑤ ς发送(σ_o, d_1, d_2,ps)到 T，请求验证配置属性(cs,ps)是否撤销。

⑥ 如果上述验证过程都通过了,ς输出 ACCEPT,否则输出 REJECT。

(5) Check

ς使用(σ_o, d_1, d_2,ps)请求 T 验证被承诺的配置属性是否被撤销,计算 $\sigma = \frac{\sigma_o}{d_1^{1/x} d_2^{1/y}}$,根据 ps 和 σ 在已经颁发的属性证书库中查找,查询到相应的项则表明被承诺的配置属性依然有效。如果 H 证明的配置属性已经被撤销,则 T 通知ς拒绝该次证明。对于 T 而言,若预先计算出 $1/x$ 和 $1/y$,则能够提高验证属性是否撤销的效率。

2. 协议安全性证明

定理 16.1(Unforgeability) 在 LRSW 假设成立、解决离散对数问题困难以及 TCM 物理安全的条件下,上述 PBA-BM 协议提供了不可伪造性的安全属性。更精确地讲就是,如果攻击者 A 能够以不可忽略的概率伪造 PBA-BM 签名,则存在模拟器 Σ 在多项式时间之内能够以不可忽略的概率破解 LRSW 假设或解决离散对数困难问题。

证明 如果攻击者 A 在属性证明过程中能够伪造 PBA-BM 签名,则将能够利用攻击者 A 构建一个算法 B 解决 LRSW 问题。下面详细阐述模拟器 S 的构造,S 按照前面所述的协议与攻击者进行攻击游戏,S 设置 T 的公钥为 $\mathrm{pk} = (q, G, g, G_T, e, X, Y, Z)$,私钥为 $\mathrm{sk} = (x, y)$,但是 S 并不知道该私钥。然后 S 将设置的公开参数告知攻击者 A。

S 模拟属性证明协议各个步骤,需要设置颁发属性证书,PBA 属性证明和属性撤销等 Random Oracle。为了实现攻击者 A 的 Oracle 查询的一致性,S 维护了下面这几种 List: L_H 用来存储用于知识签名 SPK{(cs,r_0, r, t_1, t_2) ‖ …}(…)的 Hash Oracle H 的查询和应答数据;L_I 用来保存配置属性证书颁发过程中的查询和应答记录,L_I 中的每一项记录为(cs,ps,cre,s),其中 cre=(a, A, b, B, c),s=1 表示关于(cs,ps)的配置凭证属性证书已经被撤销;反之,则为 s=0;L_S 表示 Attest 证明过程的查询和应答记录列表,L_S 中的每一项记录为(P_i, N_v,cre,σ_{PBA}, c),其中 c=1 表示证明方 P_i 已经被攻击者破解控制,也即是 P_i 平台的主机 H_i 已经被 A 所控制,反之,则 c=0。

Simulator: $H(m)$。如果$(m, h) \in L_H$,则返回 h;否则,S 选择均匀分布的随机数 $h \in_R \{0,1\}^{l_H}$,将(m, h)增加到列表 L_H 中,然后返回 h。

Simulator: Issuing(cs)。假定攻击者 A 不会进行重复的查询。给定攻击者 A 一个新的配置 cs,模拟器 S 代表可信第三方对提供的配置 cs 进行安全属性评估,设定评估的属性为 ps。模拟器 S 使用配置属性对(cs,ps)查询 Oracle O,O 应答相应的属性证书 cre=(a, A, b, B, c),然后 S 新增记录(cs,ps,cre,0)到列表 L_I。

Simulator: Revoke(cs)。假定攻击者 A 在进行配置 cs 的撤销查询之前,必须执行过同一配置的属性证书颁发查询 Issuing(cs);否则,模拟器 S 必须先执行 Issuing(cs)查询。模

拟器 S 首先在列表 L_I 查询到记录(cs，ps，cre，0)，返回 cre，然后更新该项记录为(cs，ps，cre，1)。

Simulator：Attest(cs)。设 $N_v \in \{0,1\}^{l_H}$ 是由攻击者 A 选择的随机数，$N_t \in \{0,1\}^{l_\phi}$ 则是由模拟器 S 随机选择。攻击者选择(N_v，cs，ps，cre)请求证明方 P 进行证明，S 在列表 L_I 中查询到记录(cs，ps，cre，0/1)，然后计算属性签名 σ_{PBA}，对配置进行属性证明的过程我们考虑到以下两种情况。

情况 1：安全芯片 TCM 是物理安全不可能被破解，而证明方 P(即主机 H)为诚实的证明协议参与者。S 首先计算 TCM 的配置承诺，然后根据基于属性证明协议计算 PBA-BM 签名。

(1) 攻击者 A 以随机数 N_v 挑战模拟器 S 证明，S 随机选择 $f, k \in Z_q^*$，$N_t \in \{0,1\}^{l_\phi}$，接着计算 $h_T = g_T^f$，$C = g^k$，C 是安全芯片的配置承诺，然后 TCM 对承诺签名得到 δ，S 将 g_T，h_T，N_t，C，δ 发送给主机 H。

(2) 模拟器 S 随机选择 $s_1, s_2 \in_R Z_q^*$。

(3) 模拟器 S 随机选择 $t_1, t_2, s_4, s_5 \in_R Z_q^*$，计算 $d_1 = X^{t_1}$，$d_2 = Y^{t_2}$。

(4) 模拟器 S 随机选择 $r', s_3 \in_R Z_q^*$，$c' \in_R G$，计算 $a' = a^{r'}$，$b' = b^{r'}$，$A' = A^{r'}$，$B' = B^{r'}$，$\sigma_o = \sigma \cdot g^{t_1 + t_2}$。

(5) 模拟器 S 计算 $v_x = e(X, a')$，$v_{xy} = e(X, b')$，$v_s = e(g, c')$，$v_{xyz} = e(X, B')$。

(6) 模拟器 S 随机选择 $c_H \in_R \{0,1\}^{l_H}$，在列表 L_H 中查询 c_H，如果存在则重新执行这一步。

(7) 模拟器 S 计算 $\widetilde{T}_1 = v_s^{s_3} v_{xy}^{-s_1} (v_x v_{xyz}^{ps})^{c_H - 1}$，$\widetilde{T}_2 = g_T^{s_1} h_T^{s_2} C^{c_H}$，$\widetilde{d}_1 = X^{s_4} d_1^{c_H}$，$\widetilde{d}_2 = Y^{s_5} d_2^{c_H}$。

(8) 设置 $w = q \| g \| X \| Y \| a' \| b' \| c' \| A' \| B' \| g_T \| h_T \| C \| \sigma_o \| d_1 \| d_2 \| v_x \| v_{xy} \| v_{xyz} \| v_s \| \widetilde{d}_1 \| \widetilde{d}_2 \| \widetilde{T}_1 \| \widetilde{T}_2 \| N_v \| N_t$，在列表 L_H 中查询(c_H，w)是否为其中一项记录，如果是，则回到 Attest(cs)模拟的第(1)步开始重新执行，否则，将(c_H，w)增加到列表 L_H 中。

(9) 模拟器 S 输出 $\sigma_{PBA} = (\delta, C, a', b', c', A', B', c_H, s_1, s_2, s_3, s_4, s_5)$。

(10) 模拟器 S 将(N_v，cre，σ_{PBA}，0)增加到 L_S 列表中。

情况 2：安全芯片 TCM 是物理安全不可能被破解，而证明方 P(即主机 H)却被攻击者所控制。在这种情况下，S 将模拟 A 控制主机平台 H 实际攻击 TCM 证明，如果 Attest 过程成功完成，S 将从攻击者 A 获取关于配置 cs 的属性证明签名 σ_{PBA}，否则，Attest 过程失败，也即是攻击者伪造签名失败。

对于 Attest 过程的协议输出 δ、C、a'、b'、c'、A'、B'、$\widetilde{T}_1$、$\widetilde{T}_2$、c_H、s_1、s_2、s_3、s_4、s_5，模拟器 S 的协议模拟过程与实际协议执行是不可区分的。在上述属性证明模拟过程结束后，如果攻击者 A 能够以不可忽略的概率 ε 伪造出满足验证方验证的属性证明签名，那么就能够构造出算法 B 以至少 $\frac{\varepsilon}{2}$ 解决 LRSW 假设问题，或者以至少 $\frac{\varepsilon}{2}$ 解决离散对数困难问题。给定算法 B LRSW 实例($q, G, G_T, g, g_T, e, X, Y$)，其中 $G = \langle g \rangle$ 是循环群，$X, Y \in G$，$X = g^x$，$Y = g^y$，双线性映射 $e: G \times G \to G_T$，群 G_T 的生成元为 $g_T = e(g, g)$，输入 $m \in Z_q$，算法 B 在没有以 m 查询过 LRSW Oracle O 的情况下，以不可忽略的概率输出(m, a, b, c)，满足 $a \in G$，$b = a^y$，

$c=a^{x+mxy}$，则解决 LRSW 假设问题。

假定 A 针对某平台配置属性对(cs，ps)输出伪造属性证明签名，模拟器 S 随机选择 $z\in_R Z_q$，计算 $Z=g^z$。与 LRSW 假设实例的参数设置类似，算法 B 构造 PBA-BM 方案的系统参数为 $\mathrm{pk}=(q,G,G_T,g,g_T,e,X,Y,Z)$，$\mathrm{sk}=(x,y,z)$。假定攻击者在伪造属性证明签名前共查询 q_s 次 Attest Oracle，查询的属性配置对为$(\mathrm{cs}_i,\mathrm{ps}_i)$，$i=1,\cdots,q_s$，由于 A 伪造的属性证明签名的配置属性对(cs，ps)并未查询过 Attest Oracle，所以$(\mathrm{cs},\mathrm{ps})\neq(\mathrm{cs}_i,\mathrm{ps}_i)$。下面分两种情况进行讨论属性证明协议的不可伪造性。

(1) 对于属性配置对$(\mathrm{cs}_i,\mathrm{ps}_i)$，$i=1,\cdots,q_s$，$\mathrm{cs}+\mathrm{ps}\cdot z\neq \mathrm{cs}_i+\mathrm{ps}_i\cdot z$，至少为$\frac{\varepsilon}{2}$的概率存在某个 i，满足 $\mathrm{cs}+\mathrm{ps}\cdot z=\mathrm{cs}_i+\mathrm{ps}_i\cdot z$。在这种情况下攻击者 A 可以计算得到 $z=\frac{\mathrm{cs}_i-\mathrm{cs}}{\mathrm{ps}-\mathrm{ps}_i}\bmod q$，所以将 z 设置为目标值，攻击者能够破解离散对数困难问题。

(2) 对于属性配置对$(\mathrm{cs}_i,\mathrm{ps}_i)$，$i=1,\cdots,q_s$，至少为$\frac{\varepsilon}{2}$的概率 $\mathrm{cs}+\mathrm{ps}\cdot z\neq \mathrm{cs}_i+\mathrm{ps}_i\cdot z$。若模拟器 S(扮演验证者)从攻击者 A 处获得某个有效属性证明签名 σ_{PBA}，并且该签名不在 S 保存的证明数据列表 L_S 中。S 可以重置攻击者 A 到调用证明 Oracle 产生签名中 c_H 的时间点，从而为 A 提供不同的 c_H，因此 S 可以提取出两个关于$(\delta,C,\widetilde{T}_1,\widetilde{T}_2,a',b',c',A',B')$的签名，拥有不同的 c_H 和 s_1、s_2、s_3、s_4、s_5，这两个签名设定为

$$(\delta,C,\widetilde{T}_1,\widetilde{T}_2,a',b',c',A',B',c_H^{(0)},s_1^{(0)},s_2^{(0)},s_3^{(0)},s_4^{(0)},s_5^{(0)})$$

$$(\delta,C,\widetilde{T}_1,\widetilde{T}_2,a',b',c',A',B',c_H^{(1)},s_1^{(1)},s_2^{(1)},s_3^{(1)},s_4^{(1)},s_5^{(1)})$$

设置 $a'=g^\alpha$，$b'=g^\beta$，$c'=g^\gamma$，$\alpha,\beta,\gamma\in Z_q^*$。在这两个签名中 $c_H^{(0)}\neq c_H^{(1)}$，$s_X^{(0)}\neq s_X^{(1)}$ $(X=1,2,\cdots,5)$，令 $\Delta c_H=c_H^{(0)}-c_H^{(1)}$，$\Delta s_X=s_X^{(1)}-c_X^{(0)}$。这两个签名都满足属性证明的验证条件：

$$\widetilde{T}_1=v_s^{s_3^{(0)}}v_{xy}^{-s_1^{(0)}}(v_x v_{xyz}^{\mathrm{ps}})^{c_H^{(0)}-1},\quad \widetilde{T}_1=v_s^{s_3^{(1)}}v_{xy}^{-s_1^{(1)}}(v_x v_{xyz}^{\mathrm{ps}})^{c_H^{(1)}-1}$$

其中 $v_x=e(X,a')$，$v_{xy}=e(X,b')$，$v_s=e(g,c')$，$v_{xyz}=e(X,B')$。

因此 $v_s^{\Delta s_3}=v_{xy}^{\Delta s_1}(v_x v_{xyz}^{\mathrm{ps}})^{\Delta c_H}$，由于 $\Delta c_H\neq 0$，等式两边同时进行 Δc_H^{-1} 乘方运算，再令 $\hat{s}_X=\frac{\Delta s_X}{\Delta c_H}(X=1,2,\cdots,5)$，得到 $v_s^{\hat{s}_3}=v_x v_{xy}^{\hat{s}_1}v_{xyz}^{\mathrm{ps}}$。由模拟器回绕调用(Rewind)攻击者 A 的证明过程可以知道 $\mathrm{cs}=\frac{\Delta s_1}{\Delta c_H}=\hat{s}_1$，令 $m=\hat{s}_1+\mathrm{ps}\cdot z\bmod q$，并且$(\hat{s}_1,\mathrm{ps})$为查询过证明 Oracle。由 PBA-BM 方案的验证算法可知，属性证明签名 $\sigma_{\mathrm{PBA}}=(\delta,C,a',b',c',A',B',*,*,*,*,*,*)$ 满足：(1) $e(a',Y)=e(g,b')$；(2) $v_s^{\hat{s}_3}=v_x v_{xy}^{\hat{s}_1}v_{xyz}^{\mathrm{ps}}$，即 $e(g,c')^{\hat{s}_3}=e(X,a')\cdot e(X,b')^{\hat{s}_1}\cdot e(X,B')^{\mathrm{ps}}$。

(1) $e(a',Y)=e(g,b')$，

$e(g^\alpha,g^y)=e(g,g^\beta)$

$g_T^{y\alpha}=g_T^\beta$

所以 $\beta=y\alpha\bmod q$。

(2) $e(g,c')^{\hat{s}_3}=e(X,a')\cdot e(X,b')^{\hat{s}_1}\cdot e(X,B')^{\mathrm{ps}}$

因为 $e(X,B')=e(X,A'^y)=e(X,a'^{yz})=e(X,b')^z$，所以

$$e(g,c')^{\hat{s}_3} = e(X,a') \cdot e(X,b')^{\hat{s}_1} \cdot e(X,B)^{ps}$$
$$e(g,c'^{\hat{s}_3}) = e(X,a') \cdot e(X,b)^{\hat{s}_1} \cdot e(X,b')^{ps\cdot z}$$
$$e(g,c'^{\hat{s}_3}) = e(X,a') \cdot e(X,b')^{\hat{s}_1+ps\cdot z}$$
$$e(g,g^{\hat{s}_3\cdot\gamma}) = e(g^x,g^a) \cdot e(X,b')^m$$
$$g_T^{\hat{s}_3\cdot\gamma} = g_T^{x\alpha} \cdot g_T^{mxy\alpha} = g_T^{(x+mxy)\alpha}$$

所以 $\hat{s}_3 \cdot \gamma = (x+mxy)\alpha \bmod q$。

总之,对于输入 $m = cs + ps \cdot z \bmod q$,算法 B 输出 $a = a' = g^\alpha, b = b' = g^\beta, c = c'^{\hat{s}_3} = g^{\hat{s}_3\cdot\gamma}$,则$(m,a,b,c)$就是 LRSW 假设问题的实例,它们满足:$b = g^\beta = g^{y\alpha} = a^y; c = g^{\hat{s}_3\cdot\gamma} = g^{(x+mxy)\alpha} = a^{x+mxy}$,因此,如果攻击者 A 能够以不可忽略的概率赢得攻击游戏 $\mathrm{Game}_A^{\text{att-fg}}(1^k)$,要么算法 B 就能够以不可忽略的概率解决 LRSW 问题,要么就能够以不可忽略的概率解决离散对数困难问题。

定理 16.2(Configuration Privacy) 上述 PBA-BM 协议提供了配置隐私保护的安全属性。更准确地说,如果存在攻击者 A,他能够以不可忽略的概率区分出不同的平台配置,则一定存在多项式时间内运行的模拟器 S 能够以不可忽略的概率破解承诺方案的隐蔽性。

证明 我们构建模拟器 S 扮演协议中的诚实参与方,与攻击者 A 交互进行游戏 $\mathrm{Game}_A^{\text{cf-prv}}(1^k)$。可以证明即使在攻击方 A 具有不受限的计算能力(Computationally Unbounded)的前提下,对于安全参数 k,赢得游戏的最大优势概率 $\mathrm{Adv}[A_{\text{PBA}}^{\text{cf-prv}}]$是可忽略的。如果攻击者能够攻破 PBA-BM 方案的隐私保护,则模拟器 S 就能够打开承诺中隐藏的秘密。

给定模拟器 S 的承诺 $C = g_T^{cs} h_T^{r_0}$,其中 $cs \in CS = \{cs_1, cs_2, \cdots, cs_n\}$,然后与攻击者 A 之间进行攻击游戏 $\mathrm{Game}_A^{\text{cf-prv}}(1^k)$。当 S 收到来自 A 的挑战随机数,使用 C 作为承诺执行 PBA-BM 签名协议,创建 TCM 的签名 $\delta = \mathrm{Sig}_M(C, N_v \parallel N_t)$,模拟器 S 是无法知道承诺隐藏的秘密 cs 和 r_0。因为攻击者是计算上不受限制的,S 能够计算出 α, t,满足 $h_T = g_T^\alpha, C = g_T^t = g_T^{cs+\alpha r}$,$S$ 从列表 L_1 中找出证明相同目标安全属性的配置属性证书$(cs_i, ps, cre_i)(i=1,\cdots,n)$,对于一个属性证书 cre_i 能够构造出一个元组(cs_i, r_i),满足 $t = cs_i + \alpha r_i$。模拟器按照$\mathrm{Attest}_{\text{PIK}}^{cs}$的模拟过程计算出属性证明签名 $\sigma_{\text{PBA}}^{(i)} = (\delta, C, a^{(i)'}, b^{(i)'}, c^{(i)'}, A^{(i)'}, B^{(i)'}, c_M^{(i)}, s_1^{(i)}, s_2^{(i)}, s_3^{(i)}, s_4^{(i)})$,然后 S 将 $\sigma_{\text{PBA}}^{(i)}$发送给攻击者 A。

攻击游戏结束后,攻击者 A 输出下标 j,如果 $cs_j = cs$,S 能够使用(cs_j, r_j)打开承诺 C。模拟器 S 打开承诺的概率也就是 A 判定 $cs_j = cs$ 的概率。攻击者 A 以不可忽略的概率 $\mathrm{Adv}[A_{\text{PBA}}^{\text{cf-prv}}]$赢得攻击游戏 $\mathrm{Game}_A^{\text{cf-prv}}(1^k)$,必然暗含了模拟器能够以不可忽略的概率打开承诺 C。

我们对该协议的实现性能也进行了实验分析,具体分析可参阅文献[31]。分析表明,该协议与其他属性证明协议相比,具有更短的签名长度和更少的计算量。

16.3 BCC 直接匿名证明方案

2004 年,Brickell 等人提出了直接匿名证明(Direct Anonymous Attestation,DAA)方案[32]来解决可信计算平台的匿名认证问题,以取代早期提出的 Privacy-CA 方案。之后,人

们针对该方案提出了不少扩展方案和替代方案。这方面的研究主要体现在以下3个方面。

(1) 对现有DAA方案的安全性和隐私性的改进。Camenisch提出了安全性改进方案[33],Brickell对DAA方案进行了有效的扩展,使得它能够支持更加灵活的撤销机制,Camenisch和Groth[34]利用CL-RSA签名机制来提高原始DAA方案的效率[35]。Smyth等人讨论了如何在攻陷管理员的情况下保证DAA方案的隐私性[36]。Backes等人对原始DAA方案进行了形式化的自动分析[37]。

(2) DAA方案在系统认证方面的应用。Camenisch提出了一种方案将DAA方案应用于保护匿名证书[38]。Leung和Mitchell利用DAA方案在移动普适环境中构建了一个不是基于身份的认证方案[39]。同时,人们也探讨了DAA方案在P2P系统和单点登录方面的应用。

(3) 在不同的计算环境中实现DAA。Brickell等人提出了一种基于双线性映射的DAA方案[40]。由于现有的DAA方案过于复杂,在移动平台上很难实现,因此,人们展开了对移动平台上DAA方案的研究。

本节重点介绍文献[32]中提出的直接匿名证明方案,简称为BCC方案。

16.3.1 CL-RSA签名方案

CL-RSA签名方案[35]是BCC直接匿名证明方案的基础。CL-RSA签名方案具体包括以下几个算法。

(1) 密钥生成算法。输入1^k,选择一个特殊RSA模$n=pq$,$p=2p'+1$,$q=2q'+1$。随机选择$R_0,\cdots,R_{L-1},S,Z\in QR_n$($QR_n$表示模$n$的二次剩余之集),输出公钥$(n,R_0,\cdots,R_{L-1},S,Z)$和私钥$p$、$q$。

(2) 签名算法。设ℓ_m是一个参数,那么输入$(m_0,\cdots,m_{L-1})$,其中$m_i\in\pm\{0,1\}^{\ell_m}$,即表示$m_i\in[-2^{\ell_m}+1,2^{\ell_m}-1]$,选择一个随机素数$e$,长度为$\ell_e>\ell_m+2$,以及一个随机数$v$,长度为$\ell_v=\ell_n+\ell_m+\ell_r$,其中$\ell_r$是安全参数,$\ell_n$是$n$的长度。计算$A$使得$Z\equiv R_0^{m_0}\cdots R_{L-1}^{m_{L-1}}S^vA^e(\bmod n)$,消息$(m_0,\cdots,m_{L-1})$的签名是$(e,A,v)$。

(3) 验证算法。为了验证(e,A,v)是$(m_0,\cdots,m_{L-1})$的合法签名,检查$Z\equiv R_0^{m_0}\cdots R_{L-1}^{m_{L-1}}S^vA^e(\bmod n)$,并且检查$2^{\ell_e}>e>2^{\ell_e-1}$。

定理16.3[35] CL-RSA签名方案在强RSA假设下是CCA2安全的,即抵抗适应性选择消息攻击。

16.3.2 BCC方案的安全模型

BCC方案中采用了现实系统/理想系统模型。在现实系统模型中,存在大量的参与者,他们之间可相互运行安全协议,一个敌手A可控制这些参与者中的某些,一个环境ε可给参与者u_i提供输入并可任意地与A进行交互。该环境可给诚实的参与者提供输入并收到他们的输出,可与敌手进行任意交互。不诚实的参与者都归类为敌手。在理想系统模型中,有同样的参与者,但他们不运行任何安全协议而是发送他们的输入到一个理想的可信方T并从T处收到他们的输出。可信方T从参与者的输入计算他们的输出,即应用安全协议所支持的安全功能。

所谓一个安全协议安全地实现了一个安全功能是指,如果对每一个敌手A和每一个环

境 ε 都存在一个模拟器 S，S 在理想系统中所控制的参与者与 A 在现实系统中所控制的参与者相同，并且环境 ε 不能区分它自己是运行在与 A 交互的现实系统中还是在与 S 交互的理想系统中。

BCC 方案中的主要参与者包括：一个颁发者 I，一个具有身份 id_i 的可信平台模块(TPM)M_i，一个带有 TPM M_i 的主机 H_i，一个假冒检测预言器 O 和一个验证者 V_i。

理想系统模型中的可信方 T 支持以下操作。

(1) Setup：每个参与者与 T 交互，表明该参与者是否已经被攻击方攻陷(Corrupted)。每个 TPM M_i 发送它的唯一身份 id_i 给 T，T 将其转交给相应的主机 H_i。

(2) Join：可信计算平台 H_i 向 T 发出请求，希望成为群成员，T 询问 M_i 是否希望成为群的一员，如果 M_i 同意，T 向颁发者 I 发送消息表明身份为 id_i 的可信计算平台希望加入，如果 M_i 是假冒的，T 将向颁发者 I 表明这一点。如果 I 批准，T 向 H_i 通知其已经成功地加入。

(3) DAA-Sign/Verify：H_i 拟对消息 m 进行签名，使用计数器值 cnt 和基名 bsn$\in\{0,1\}^*\cup\{\bot\}$。H_i 将 m、bsn 和 cnt 发送给 T。首先 T 验证 H_i/M_i 是否为群的成员，如果不是，T 将拒绝 H_i/M_i 的请求；否则，T 将 m 和 cnt 转交给相应的 M_i，询问是否同意签名。如果 M_i 同意，T 向 H_i 表明 M_i 同意签名并询问 H_i 是否需要签名。如果 H_i 没有退出，T 执行以下步骤。

① 如果 M_i 是假冒的，T 通知 V_j：假冒 TPM 对 m 进行了签名。

② 如果 bsn$=\bot$，T 通知 V_j：H_i/M_i 已经用 bsn 对 m 进行了签名。

③ 如果 bsn$\neq\bot$，T 检查 H_i/M_i 是否已经用参数 bsn 和 cnt 对消息进行了签名。如果是，T 在它的假名数据库中查找对应的假名 P，如果不是，将随机生成一个假名 $P\in_R\{0,1\}^{l_\sigma}$(l_σ 是一个安全参数)，T 通知 V_j：假名为 P 的平台对消息 m 进行了签名。

④ Rogue Tagging：假冒检测预言器 O 告诉 T 具有身份 id 的平台是假冒的。如果具有身份 id 的 TPM 没有被攻陷，T 拒绝请求；否则，T 将具有身份 id 的 TPM 标记为假冒。

该理想系统模型具有以下的安全特性。

① 不可伪造性(Unforgeability)：只有群成员且不是检测假冒的 TPM 或平台才能进行签名操作，也就是说不是群成员的用户或者已经被撤销的群成员不能成功地进行签名操作。

② 匿名性(Anonymity)：验证者不能标识出签名者的身份，如果 bsn$=\bot$，签名是完全匿名的，如果 bsn$\neq\bot$，签名具有部分的匿名性，验证者通过假名 P 标识签名者。

③ 不可关联性(Unlinkability)：如果 bsn$=\bot$，验证者无法区分两个不同的签名是否由同一个可信计算平台签发。

假定没有两个验证者使用同一个基名 bsn，如果有这样的情况发生，则认为他们是同一个验证者。当然，一个验证者可以使用多个不同的基名。也假定 TPM 有唯一的身份 id 可以识别自身，然而，颁发者总是可以加一些新的 Host/TPM 到系统，因此 Host/TPM 的数量是不固定的。在 BCC 方案的安全性证明中，也假定敌手总是控制着假冒检测预言器 O，并且不考虑被攻陷的 TPM 嵌入诚实的主机(Host)的情形。

16.3.3 BCC 方案的基本思想

BCC 方案的基本思想与 CL-RSA 匿名证书系统的思想类似[35]：TPM 选择一个秘密的

消息 f，从颁发者(Issuer)那里通过安全的两方协议获得该消息 f 的 CL-RSA 签名，然后 TPM 可以匿名地向验证者证明 TPM 拥有对秘密消息 f 的签名。为了检测出假冒 TPM，TPM 必须向验证者发送假名 N_V，同时证明 N_V 是通过秘密消息 f 计算出来的。

概括来讲，BCC 方案主要包括了以下几个参与者。

(1) DAA 颁发者(DAA Issuer)，该参与者是发布签名的机构。

(2) 可信计算平台(Host/TPM)，该参与者是 TPM 或带有 TPM 的 Host。

(3) 验证者(Verifier)，验证签名的参与者。

为了简单起见，BCC 方案的安全模型中没有模型化假冒检测预言器 O，因为一个诚实的验证者原则上是在假冒检测之前可以识别假冒 TPM 的签名，因此，一般不考虑这一参与者。

DAA 颁发者的参数建立过程如下：

(1) DAA 颁发者选择一个 RSA 模数 $n=pq$，$p=2p'+1$，$q=2q'+1$，p,p',q,q'都是素数，n 的比特长度为 l_n。

(2) DAA 颁发者选择一个随机生成元 $g'\in QR_n$(模 n 的二次剩余群)。

(3) DAA 颁发者选择随机整数 $x_0,x_1,x_z,x_s,x_h,x_g\in[1,p'q']$，并计算

$$g=(g')^{x_g}\bmod n,\quad h=(g')^{x_h}\bmod n,\quad S=h^{x_s}\bmod n,$$
$$Z=h^{x_z}\bmod n,\quad R_0=S^{x_0}\bmod n,\quad R_1=S^{x_1}\bmod n$$

(4) DAA 颁发者生成一个非交互零知识证明协议，并证明 R_0、R_1、S、Z、g、h 是按照正确方式产生的，即 $g,h\in\langle g'\rangle$，$S,Z\in\langle h\rangle$，$R_0,R_1\in\langle S\rangle$。文献[32]使用了以下的零知识证明协议。

① 证明者选择随机数。

$$\tilde{x}_{(g,1)},\cdots,\tilde{x}_{(g,\ell_H)}\in_R[1,p'q'],\quad \tilde{x}_{(h,1)},\cdots,\tilde{x}_{(h,\ell_H)}\in_R[1,p'q'],$$
$$\tilde{x}_{(s,1)},\cdots,\tilde{x}_{(s,\ell_H)}\in_R[1,p'q'],\quad \tilde{x}_{(z,1)},\cdots,\tilde{x}_{(z,\ell_H)}\in_R[1,p'q'],$$
$$\tilde{x}_{(0,1)},\cdots,\tilde{x}_{(0,\ell_H)}\in_R[1,p'q'],\quad \tilde{x}_{(1,1)},\cdots,\tilde{x}_{(1,\ell_H)}\in_R[1,p'q']$$

并对 $i=1,\cdots,\ell_H$ 计算

$$\tilde{g}_{(g,i)}:=(g')^{\tilde{x}_{(g,i)}}\bmod n,\quad \tilde{h}_{(h,i)}:=(g')^{\tilde{x}_{(h,i)}}\bmod n,\quad \tilde{S}_{(s,i)}:=h^{\tilde{x}_{(s,i)}}\bmod n,$$
$$\tilde{Z}_{(z,i)}:=h^{\tilde{x}_{(z,i)}}\bmod n,\quad \tilde{R}_{(0,i)}:=S^{\tilde{x}_{(0,i)}}\bmod n,\quad \tilde{R}_{(1,i)}:=S^{\tilde{x}_{(1,i)}}\bmod n$$

② 证明者计算。

$$c:=H(n,g,g',h,S,Z,R_0,R_1,\tilde{g}_{(g,1)},\cdots,\tilde{g}_{(g,\ell_H)},\tilde{h}_{(h,1)},\cdots,\tilde{h}_{(h,\ell_H)},\tilde{S}_{(s,1)},\cdots,$$
$$\tilde{S}_{(s,\ell_H)},\tilde{Z}_{(z,1)},\cdots,\tilde{Z}_{(z,\ell_H)},\tilde{R}_{(0,1)},\cdots,\tilde{R}_{(0,\ell_H)},\tilde{R}_{(1,1)},\cdots,\tilde{R}_{(1,\ell_H)})$$

③ 证明者对 $i=1,\cdots,\ell_H$ 计算。

$$\hat{x}_{(g,i)}=\tilde{x}_{(g,i)}-c_ix_g\bmod p'q',\quad \hat{x}_{(h,i)}=\tilde{x}_{(h,i)}-c_ix_h\bmod p'q',$$
$$\hat{x}_{(s,i)}=\tilde{x}_{(s,i)}-c_ix_s\bmod p'q',\quad \hat{x}_{(z,i)}=\tilde{x}_{(z,i)}-c_ix_z\bmod p'q',$$
$$\hat{x}_{(0,i)}=\tilde{x}_{(0,i)}-c_ix_0\bmod p'q',\quad \hat{x}_{(1,i)}=\tilde{x}_{(1,i)}-c_ix_1\bmod p'q'$$

其中 c_i 是 c 的第 i 比特。

④ 证明者公布。

$$\text{proof}:=(c,\hat{x}_{(g,1)},\cdots,\hat{x}_{(g,\ell_H)},\hat{x}_{(h,1)},\cdots,\hat{x}_{(h,\ell_H)},\hat{x}_{(s,1)},\cdots,\hat{x}_{(s,\ell_H)},\hat{x}_{(z,1)},\cdots,$$

$\hat{x}_{(z,\ell_H)},\hat{x}_{(0,1)},\cdots,\hat{x}_{(0,\ell_H)},\hat{x}_{(1,1)},\cdots,\hat{x}_{(1,\ell_H)})$

作为 $g,h\in\langle g'\rangle,S,Z\in\langle h\rangle,R_0,R_1\in\langle S\rangle$的证明。

(5) DAA 颁发者生成一个素数阶群。随机选择素数 ρ 和 Γ 使得 $\Gamma=r\rho+1$,其中 r 不能被 ρ 除尽,并且 $2^{\ell_\Gamma-1}<\Gamma<2^{\ell_\Gamma}$,$2^{\ell_\rho-1}<\rho<2^{\ell_\rho}$。选择一个随机数 $\gamma'\in_R Z_\Gamma^*$ 使得 $\gamma'^{(\Gamma-1)/\rho}\neq 1(\mathrm{mod}\ \Gamma)$,并且令 $\gamma=\gamma'^{(\Gamma-1)/\rho}\mathrm{mod}\ \Gamma$。

(6) DAA 颁发者发布公钥$(n,g',g,h,S,Z,R_0,R_1,\gamma,\Gamma,\rho)$和非交互零知识证明协议,并把 $p'q'$作为秘密密钥保存。

设 $H_\Gamma(\cdot):\{0,1\}^*\to\{0,1\}^{\ell_\Gamma+\ell_\Phi}$和 $H(\cdot):\{0,1\}^*\to\{0,1\}^{\ell_H}$是两个抗碰撞的 Hash 函数,其中 l_Γ 是模 Γ 的比特长度,l_Φ 是控制统计零知识特性的安全参数的比特长度,l_H 是用于 Fiat-Shamir 启发式签名算法的 Hash 函数的输出长度。

DAA 颁布者的公钥的验证过程如下。

(1) 验证 $g,h\in\langle g'\rangle,S,Z\in\langle h\rangle,R_0,R_1\in\langle S\rangle$,这一步与 DAA 颁发者的参数建立过程的第(4)步对应,具体过程如下。

① 对 $i=1,\cdots,\ell_H$ 计算:

$$\hat{g}_{(g,i)}:=g^{c_i}g'^{\hat{x}_{(g,i)}}\mathrm{mod}\ n,\quad \hat{h}_{(h,i)}:=h^{c_i}g'^{\hat{x}_{(h,i)}}\mathrm{mod}\ n,\quad \hat{S}_{(s,i)}:=S^{c_i}h^{\hat{x}_{(s,i)}}\mathrm{mod}\ n;$$

$$\hat{Z}_{(z,i)}:=Z^{c_i}h^{\hat{x}_{(z,i)}}\mathrm{mod}\ n,\quad \hat{R}_{(0,i)}:=R_0^{c_i}S^{\hat{x}_{(0,i)}}\mathrm{mod}\ n,\quad \hat{R}_{(1,i)}:=R_1^{c_i}S^{\hat{x}_{(1,i)}}\mathrm{mod}\ n$$

其中 c_i 是 c 的第 i 比特。

② 验证是否有:

$$c:=H(n,g,g',h,S,Z,R_0,R_1,\hat{g}_{(g,1)},\cdots,\hat{g}_{(g,\ell_H)},\hat{h}_{(h,1)},\cdots,\hat{h}_{(h,\ell_H)},\hat{S}_{(s,1)},\cdots,$$

$$\hat{S}_{(s,\ell_H)},\hat{Z}_{(z,1)},\cdots,\hat{Z}_{(z,\ell_H)},\hat{R}_{(0,1)},\cdots,\hat{R}_{(0,\ell_H)},\hat{R}_{(1,1)},\cdots,\hat{R}_{(1,\ell_H)})$$

(2) 检查 ρ 和 Γ 是否为素数,是否有 $\rho|(\Gamma-1)$,ρ 不能除尽$(\Gamma-1)/\rho$ 和 $\gamma^\rho\equiv1(\mathrm{mod}\ \Gamma)$。

(3) 检查所有的公钥参数是否都具有所要求的长度。

BCC 方案主要执行下面几个协议或操作。

(1) DAA-Join 协议:该协议的参与方是 DAA 颁发者和可信计算平台,首先可信计算平台生成一个秘密消息 f,然后将其拆分成两个长为 l_f 比特的秘密消息 f_1 和 f_2。接下来利用两方协议对秘密消息进行签名,签名过程如下:首先 TPM 计算并发送消息对(f_0,f_1)的承诺 $U=R_0^{f_0}R_1^{f_1}S^{v'}$ 和 $N_I=\varsigma_I^{f_0+f_1 2^{\ell_f}}\mathrm{mod}\ \Gamma$ 给 DAA 颁发者,其中 v'是 TPM 随机选择的盲因子,ς_I 是由 DAA 的名字导出的值;然后通过零知识证明协议证明 U 和 N_I 被正确地构造。DAA 颁发者随机选择一个整数 v''和一个素数 e,并计算 $A=(Z/(US^{v''}))^{1/e}\mathrm{mod}\ n$,将$(A,e,v'')$发送给 TPM,并向 TPM 证明 A 被正确地构造,那么 TPM 可得到(f_0,f_1)的签名就是$(A,e,v=v'+v'')$。此时,TPM 可利用知识的零知识证明协议证明他拥有 f_0 和 f_1 的 CL-RSA 签名。

(2) DAA-Sign 操作:在这个操作中,可信计算平台用秘密消息(f_0,f_1)和(A,e,v)对 TPM 产生的 AIK(即身份证明密钥)公钥进行知识签名。

(3) DAA-Verify 操作:在这个操作中,验证者验证签名的合法性。

16.3.4　BCC 方案的具体协议

1. DAA-Join 协议

设 $\mathrm{PK}_I=(n,g',g,h,S,Z,R_0,R_1,\gamma,\Gamma,\rho)$是 DAA 颁发者的公钥,$\mathrm{PK}_I'$是 DAA 颁发者用

于认证 PK_I 的长期公钥。设 $\zeta_I = (H_\Gamma(1 \| bsn_I))^{(\Gamma-1)/\rho} \bmod \Gamma$，其中 bsn_I 是 DAA 颁发者的长期基名。假定在运行 DAA-Join 协议之前，主机和 TPM 都已经使用 PK'_I 认证了 PK_I。

设 DAAseed 是 TPM 用于计算 f_0 和 f_1 的秘密种子，cnt 是计数器的当前值，该计数器用于记录 TPM 已运行 DAA-Join 协议的次数。假定 TPM 和 DAA 颁发者之间已经建立了一个单向认证信道，即 DAA 颁发者需要确信他正在与正确的 TPM 交谈。这可由下列方式确立。

(1) DAA 颁发者选择一个随机数 $n_e \in \{0,1\}^{\ell_\phi}$，使用 TPM 的背书公钥 EK 加密 n_e 后发送给 TPM。

(2) TPM 解密密文后获得 n_e，然后 TPM 计算 $a_U := H(U \| n_e)$，并把 a_U 发送给 DAA 颁发者。

(3) DAA 颁发者验证是否有 $a_U = H(U \| n_e)$ 成立。

DAA-Siign 协议的具体过程如下。

(1) 主机 Host 计算 $\zeta_I := (H_\Gamma(1 \| bsn_I))^{(\Gamma-1)/\rho} \bmod \Gamma$，并发送 ζ_I 给 TPM。

(2) TPM 检查是否有 $\zeta_I^\rho \equiv 1 \pmod \Gamma$。设 $i := \left\lfloor \frac{\ell_\rho + \ell_\phi}{\ell_H} \right\rfloor$，TPM 计算 $f := H(H(\text{DAAseed} \| H(PK'_I) \| \text{cnt} \| 0) \| \cdots \| H(\text{DAAseed} \| H(PK_I') \| \text{cnt} \| i)$，$f_0 := \text{LSB}_{l_f}(f)$，$f_1 := \text{CAR}_{l_f}(f)$，$v' \in_R \{0,1\}^{l_n + l_\phi}$，$U := R_0^{f_0} R_1^{f_1} S^{v'} \bmod n$，$N_I := \zeta_I^{f_0 + f_1 2^{\ell_f}} \bmod \Gamma$，并发送 U 和 N_I 给主机，由主机转发给 DAA 颁发者。

(3) DAA 颁发者检查所有在黑名单上的 f_0、f_1 是否有 $N_I \equiv \zeta_I^{f_0 + f_1 2^{\ell_f}} \bmod \Gamma$，DAA 颁发者还检查这个平台是否以前用过 N_I，如果 DAA 颁发者发现它在黑名单上，就终止协议。

(4) TPM 向 DAA 颁发者证明拥有 f_0、f_1 和 v'，它作为证明者向作为验证者的 DAA 颁发者执行协议

$$\text{SPK}\{(f_0, f_1, v'): U = \pm R_0^{f_0} R_1^{f_1} S^{v'} \bmod n \wedge N_I := \zeta_I^{f_0 + f_1 2^{\ell_f}} \bmod \Gamma \wedge f_0,$$
$$f_1 \in \{0,1\}^{\ell_f + \ell_\phi + \ell_H + 2} \wedge v' \in \{0,1\}^{\ell_n + \ell_\phi + \ell_H + 2}\}(n_t \| n_i)$$

具体协议如下。

① TPM 选择随机整数 $r_{f_0}, r_{f_1} \in_R \{0,1\}^{\ell_f + \ell_\phi + \ell_H}$ 和 $r_{v'} \in_R \{0,1\}^{\ell_f + 2\ell_\phi + \ell_H}$，计算 $\tilde{U} := R_0^{f_0} R_1^{f_1} S^{v'} \bmod n$ 和 $\tilde{N}_I := \zeta_I^{f_0 + f_1 2^{\ell_f}} \bmod n$，并发送 $\tilde{U}$ 和 $\tilde{N}_I$ 给主机。

② DAA 颁发者选择一个随机串 $n_i \in \{0,1\}^{\ell_H}$，并把 n_i 发送给主机。

③ 主机计算 $c_h := H(n \| R_0 \| R_1 \| S \| U \| N_I \| \tilde{U} \| \tilde{N}_I \| n_i)$，并把 c_h 发送给 TPM。

④ TPM 选择一个随机数 $n_t \in \{0,1\}^{\ell_\phi}$，并计算 $c := H(c_t \| n_t) \in [0, 2^{\ell_H} - 1]$。

⑤ TPM 计算 $s_{f_0} = r_{f_0} + c \cdot f_0$，$s_{f_1} = r_{f_1} + c \cdot f_1$ 和 $s_{v'} = r_{v'} + c \cdot v$，并把 $(c, n_t, s_{f_0}, s_{f_1}, s_{v'})$ 发送给主机。

⑥ 主机把 $(c, n_t, s_{f_0}, s_{f_1}, s_{v'})$ 转发给 DAA 颁发者。

⑦ DAA 颁发者验证证明，通过计算 $\hat{U} = U^{-c} R_0^{f_0} R_1^{f_1} S^{v'} \bmod n$，$\hat{N}_I := N_I^{-c} \zeta_I^{s_{f_0} + s_{f_1} 2^{\ell_f}} \bmod \Gamma$，并检查是否有 $c_h = H(H(n \| R_0 \| R_1 \| S \| U \| N_I \| \hat{U} \| \hat{N}_I \| n_i) \| n_t)$，$s_{f_0}, s_{f_1} \in \{0,1\}^{\ell_f + \ell_\phi + \ell_H + 1}$，$s_{v'} \in \{0,1\}^{\ell_n + 2\ell_\phi + \ell_H + 1}$。

(5) DAA 颁发者选择一个随机数 $\hat{v} \in_R \{0,1\}^{\ell_v - 1}$ 和一个素数 $e \in_R \{2^{\ell_e - 1}, 2^{\ell_e - 1} + 2^{\ell'_e - 1}\}$，

并计算 $v'':=\hat{v}+2^{\ell_v-1}$ 和 $A:=\left(\frac{Z}{US^{v''}}\right)^{1/e}\bmod n$。

(6) 为使主机相信 A 是正确计算的，DAA 颁发者作为一个证明者和主机运行协议 $\mathrm{SPK}\left\{(d):A:=\pm\left(\frac{Z}{US^{v''}}\right)^{d}\bmod n\right\}(n_h)$。

① 主机选取一个随机整数 $n_h\in\{0,1\}^{\ell_\phi}$，并把 n_h 发送给 DAA 颁发者。

② DAA 颁发者随机选取 $r_e\in_R\{0,p'q'\}$，计算 $\tilde{A}:=\left(\frac{Z}{US^{v''}}\right)^{r_e}\bmod n$，$c'=H(n\|Z\|S\|U\|v''\|A\|\tilde{A}\|n_h)$ 和 $s_3=r_e+c'/e\bmod p'q'$，并把 c'、s_e 和 (A,e,v'') 发送给主机。

③ 主机验证是否 e 是一个素数并且属于 $[2^{\ell_e-1},2^{\ell_e-1}+2^{\ell_e'-1}]$，计算 $\hat{A}:=A^{c'}\left(\frac{Z}{US^{v''}}\right)^{s_e}\bmod n$，并检查是否有 $c'=H(n\|Z\|S\|U\|v''\|A\|\hat{A}\|n_h)$。

(7) 主机转发 v'' 给 TPM。

(8) TPM 收到 v'' 后，设置 $v:=v'+v''$，并存储 (f_0,f_1,v)。

DAA-Sing 协议运行结束后，TPM 将获得秘密值 f_0、f_1 和 v，主机将获得值 A 和 e，DAA 颁发者将获得值 N_I，这些值满足条件 $A^eR_0^{f_0}R_1^{f_1}S^v\equiv Z(\bmod n)$ 和 $N_I\equiv\varsigma_I^{f_0+f_1 2^{\ell_f}}(\bmod \Gamma)$。

2. DAA-Sign 协议

设 $n_v\in\{0,1\}^{l_H}$ 是一个 nonce 值，bsn_v 是一个由验证者提供的基名值。设 b 是一个描述协议使用情况的字节，即如果 $b=0$，则说明消息 m 是由 TPM 产生的；如果 $b=1$，则说明消息 m 是 TPM 的输入。

(1) ① 依据验证方的请求(即 $\mathrm{bsn}_v\neq\perp$ 或 $\mathrm{bsn}_v=\perp$)，主机计算 ζ 的值。

$\zeta\in_R\langle\gamma\rangle$ 或 $\zeta:=(H_\Gamma(1\|\mathrm{bsn}_V))^{(\Gamma-1)/\rho}\bmod\Gamma$，并把 ζ 发送给 TPM。

② TPM 检查是否有 $\zeta^\rho\equiv1(\bmod\Gamma)$。

(2) ① 主机选择随机整数 $w,r\in_R\{0,1\}^{l_n+l_\phi}$，并计算 $T_1:=Ah^w\bmod n$ 和 $T_2:=g^wh^e(g')^r\bmod n$。

② TPM 计算 $N_V:=\zeta_I^{f_0+f_1 2^{\ell_f}}\bmod\Gamma$ 并发送 N_V 给主机。

(3) TPM 和主机共同产生一个“知识签名”，即 T_1 和 T_2 对证书承诺，并且 N_V 是从与这个证书相关联的密钥计算得到的，即计算“知识签名”。

$$
\begin{aligned}
&\mathrm{SPK}\{(f_0,f_1,v,e,w,r,ew,ee,er):\\
&Z\equiv\pm T_1^eR_0^{f_0}R_1^{f_1}S^vh^{-ew}(\bmod n)\wedge T_2\equiv\pm g^wh^{-ew}g'^r(\bmod n)\wedge 1\\
&\quad\equiv\pm T_2^{-e}g^{ew}h^{ee}g'^{er}(\bmod n)\\
&\quad\wedge N_V:=\zeta^{f_0+f_1 2^{\ell_f}}\bmod\Gamma\wedge f_0,f_1\in\{0,1\}^{\ell_f+\ell_\phi+\ell_H+2}\\
&\quad\wedge(e-2^{l_e})\in\{0,1\}^{\ell_e'+\ell_\phi+\ell_H+1}\}(n_t\|n_v\|b\|m)
\end{aligned}
$$

实际上主机知道大部分的秘密值，只有 f_0、f_1 和 v 需要由 TPM 来计算，计算过程如下。

① a. TPM 选取随机整数 $r_v\in_R\{0,1\}^{\ell_v+\ell_\phi+\ell_H}$ 和 $r_{f_0},r_{f_1}\in_R\{0,1\}^{\ell_f+\ell_\phi+\ell_H}$，计算 $\tilde{T}_{1t}:=R_0^{r_{f_0}}R_1^{r_{f_1}}S^{r_v}(\bmod n)$，$\tilde{r}_f=r_{f_0}+r_{f_1}2^{\ell_f}$，$\hat{N}_V:=\zeta^{\tilde{r}_f}\bmod\Gamma$，并发送 $\tilde{T}_{1t}$ 和 $\hat{N}_V$ 给主机。

b. 主机选择随机数 $r_e\in_R\{0,1\}^{\ell_e'+\ell_\phi+\ell_H}$，$r_{ee}\in_R\{0,1\}^{2\ell_e+\ell_\phi+\ell_H+1}$，$r_w,r_r\in_R\{0,1\}^{\ell_n+2\ell_\phi+\ell_H}$，

$r_{ew}, r_{er} \in_R \{0,1\}^{\ell_e+\ell_n+2\ell_\phi+\ell_H+1}$，并计算 $\widetilde{T}:=\widetilde{T}_{1t}\widetilde{T}_1^{r_e}h^{-r_{ew}}(\bmod\ n)$，$\widetilde{T}_2:=g^{r_w}h^{r_e}(g')^{r_r}\bmod n$，$\widetilde{T}_2':=\widetilde{T}_2^{-r_e}g^{r_{ew}}h^{r_{ee}}(g')^{r_{er}}\bmod n$。

② a. 主机计算

$c_h=H(H(n\|g\|g'\|h\|R_0\|R_1\|S\|Z\|\gamma\|\Gamma\|\rho)\|\zeta\|(T_1\|T_2)\|N_V\|(\widetilde{T}_1\|\widetilde{T}_2\|\widetilde{T}_2')\|\widetilde{N}_V)\|n_v)\in[0,2^{\ell_H-1}]$，并把 c_h 发送给 TPM。

b. TPM 选择随机数 $n_t\in\{0,1\}^{\ell_\phi}$，计算 $c:=H(H(c_h\|n_t)\|b\|m)$，并把 c 和 n_t 发送给主机。

③ a. TPM 计算(整数域)$s_v=r_v+c\cdot v$，$s_{f_0}=r_{f_0}+c\cdot f_0$，$s_{f_1}=r_{f_1}+c\cdot f_1$，并把$(s_v,s_{f_0},s_{f_1})$发送给主机。

b. 主机计算(整数域)$s_e:=r_e+c\cdot(e-2^{\ell_e-1})$，$s_{ee}:=r_{ee}+c\cdot e^2$，$s_w:=r_w+c\cdot w$，$s_w:=r_w+c\cdot w\cdot e$，$s_r:=r_r+c\cdot r$，$s_{er}:=r_{er}+c\cdot e\cdot r$。

(4) 主机输出消息 m 的签名 $\sigma:=(\zeta,(T_1,T_2),N_V,c,n_t,(s_v,s_{f_0},s_{f_1},s_e,s_{ee},s_w,s_{ew},s_r,s_{er}))$。

3. DAA-Verify 协议

对消息 m 的签名 $\sigma=(\zeta,(T_1,T_2),N_V,c,n_t,(s_v,s_{f_0},s_{f_1},s_e,s_{ee},s_w,s_{ew},s_r,s_{er}))$的验证过程如下(公钥为$(n,g',g,h,S,Z,R_0,R_1,\gamma,\Gamma,\rho)$)。

(1) 计算

$$\hat{T}_1:=Z^{-c}T_1^{s_e+c2^{\ell_e-1}}R_0^{s_{f_0}}R_1^{s_{f_1}}S^{s_v}h^{-s_{ew}}(\bmod\ n),$$

$$\hat{T}_2':=T_2^{-(s_e+c2^{\ell_e-1})}g^{s_{ew}}h^{s_{ee}}g'^{s_{er}}\bmod n,\quad \hat{T}_2:=T_2^{-c}g^{s_w}h^{s_e+c2^{\ell_e-1}}g'^{s_r}\bmod n,$$

$$\hat{N}_V:=N_V^{-c}\zeta^{f_0+f_1 2^{\ell_f}}\bmod \Gamma。$$

(2) 验证是否有

$c=H(H(H(n\|g\|g'\|h\|R_0\|R_1\|S\|Z\|\gamma\|\Gamma\|\rho)\|\zeta\|(T_1\|T_2)\|N_V\|(\hat{T}_1\|\hat{T}_2\|\hat{T}_2')\|\hat{N}_V\|n_v)\|n_t)\|b\|m)$ $N_V,\zeta\in\langle\gamma\rangle$(可通过计算 N_v^ρ 和 ξ^ρ 是否全为 1 来验证，其中 ρ 是 γ 的阶)，$s_{f_0},s_{f_1}\in\{0,1\}^{\ell_f+\ell_\phi+\ell_H+1}$，$s_e\in\{0,1\}^{\ell_e'+\ell_\phi+\ell_H+1}$。

(3) 如果 ζ 是由验证者的基名派生的，则检查是否有 $\zeta\equiv(H_\Gamma(1\|\mathrm{bsn}_V))^{(\Gamma-1)/\rho}\bmod \Gamma$。

(4) 对所有在黑名单上的 f_0、f_1，检查是否有 $N_V\equiv\zeta^{f_0+f_1 2^{\ell_f}}\bmod \Gamma$。

4. 假冒检测

当一个证书(A,e,v)和秘密值 f_0、f_1 被发现在 Internet 上或嵌入到某个软件中，则应该将他们发送到所有可能的验证者手中。这些验证者验证是否有 $A^eR_0^{f_0}R_1^{f_1}S^v\equiv Z(\bmod\ n)$成立，并把 f_0、f_1 放在假冒密钥的黑名单里。这里并不需要一个证书撤销权威机构。

关于 BCC 方案的安全性已有以下结论。

定理 16.4[32]　在$<\gamma>$群的 DDH 假设和强 RSA 假设下，BCC 方案实现了一个安全的直接匿名证明系统，即满足不可伪造性、匿名性和不可关联性。

16.4 跨域直接匿名证明方案

在 BCC 方案中，主要的参与方有 DAA 颁发者(Issuer)，可信计算平台(Host/TPM)，验证者(Verifier)。其中 DAA 颁发者是一个发布 DAA 签名的权威机构，各个不同的 TPM 厂商都设置有自己的 DAA 颁发者，这样就形成了相对独立的信任域，不同的信任域会有不同的 DAA 颁发者，不同信任域内的参与者将信任不同的 DAA 颁发者。而 BCC 方案只适用于单信任域的情况，当位于不同域的验证者和可信计算平台需要交互时，由于验证者和可信计算平台信任不同的 DAA 颁发者，将不能进行正常的直接匿名证明。例如，位于信任域 *A* 的验证者 *A* 只信任公司 *A* 的 DAA 颁发者，而不信任公司 *B* 的 DAA 颁发者，即使可信计算平台 *B* 得到了公司 *B* 的 DAA 颁发者签发的证书，公司 *A* 的验证者也不能信任可信计算平台 *B*。

针对 BCC 方案的这一缺陷，我们提出了跨域 DAA 方案。该方案利用零知识证明协议，有效地解决了跨信任域的直接匿名证明问题，为各个不同厂商的可信计算平台之间的匿名通信建立了基础，而且该方案不需要修改现有的 TPM 规范，可以直接在 TPM 1.2 上实现。另外，对跨域 DAA 方案的匿名性和不可伪造性进行了详细的形式化证明。

16.4.1 跨域 DAA 系统结构

在本节介绍的跨域 DAA 系统(Inter-Domain DAA，IDAA)中，假设有两个信任域 Domain *A* 和 Domain *B*，在 BCC 方案的基础上，增加两个参与方——护照颁发者(Passport Issuer)和签证颁发者(Visa Issuer)。护照颁发者向申请者颁发护照证书(Passport Certificate)，而签证颁发者向申请者颁发签证证书(Visa Certificate)。

跨域 DAA 的基本思想是：如果 Domain *A* 的可信计算平台 *A*(Host/TPM A)要向 Domain *B* 的验证者 *B*(Verifier *B*)证明可信计算平台 *A* 的身份(假设 *A* 已经得到 DAA 颁发者的签名)，同时不暴露自己的隐私，那么首先可信计算平台 *A* 向本地的护照颁发者申请一个护照证书，该护照证书证明了可信计算平台 *A* 在信任域 *A* 中的身份。然后可信计算平台 *A* 用申请到的护照证书向信任域 *B* 的签证颁发者申请签证证书。最后，可信计算平台 *A* 用护照证书和签证证书向信任域 *B* 中的验证者 *B* 匿名地证明可信计算平台 *A* 的身份。该结构可以很方便地扩展到多个信任域的情形。跨域 DAA 系统结构如图 16.8 所示。

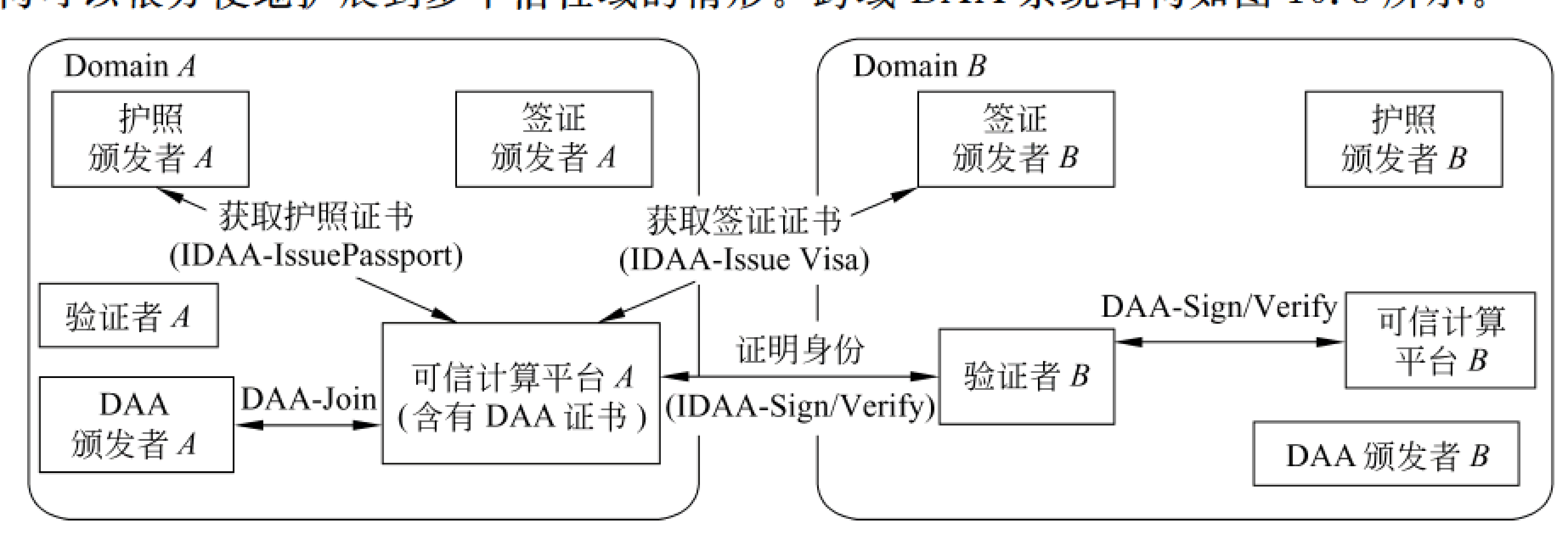


图 16.8 跨域 DAA 系统结构

在 BCC 方案的基础上，跨域 DAA 系统增加了以下几个协议或操作。

(1) IDAA-IssuePassport 协议：通过该协议 Domain A 的护照颁发者 A 向 Domain A 的可信计算平台 A 签发护照证书。

(2) IDAA-IssueVisa 协议：通过该协议 Domain B 的签证颁发者 B 向 Domain A 的可信计算平台 A 签发签证证书。签发过程中需要匿名地使用护照证书和 TPM 的秘密消息 f_0、f_1。

(3) IDAA-Sign/IDAA-Verify 操作：通过该操作 Domain A 的可信计算平台 A 向 Domain B 的验证者 B 匿名地证明身份。只有同时拥有护照证书和签证证书的可信计算平台才能向不同域的验证者证明自己的身份。

在下面的讨论中，ℓ_ϕ、ℓ_n 等为安全参数，安全参数的选择与 BCC 方案中的一致。

16.4.2 跨域 DAA 安全模型

下面将描述跨域 DAA 方案的理想系统的可信方 T。跨域 DAA 的可信方 T 在 BCC 方案的可信方的基础上构建[32]，并且增加了以下几个操作。

(1) IDAA-PassportJoin：H_i 向 T 发出请求，希望成为 Passport 群体的一员，H_i 的计数器值是 cnt，然后 T 向 TPM M_i 发送 cnt，并询问 M_i 是否希望成为 Passport 群体的一员，如果 M_i 同意，那么 T 检查 H_i/M_i 是否已经成功执行了 DAA-Join 操作。如果不是，T 将拒绝其请求；如果是，T 向护照颁发者(Passport Issuer)PI 询问具有身份 id 以及计数器为 cnt 的平台是否能成为 Passport 群体的一员，如果 M_i 是假冒的，T 将向护照颁发者声明这一点。如果护照颁发者批准，T 向 H_i 通知其已经成功地加入。

(2) IDAA-VisaJoin：其过程与 IDAA-PassportJoin 类似。

(3) DAA-Sign/Verify：H_i 希望对消息 m 进行跨域签名，验证者是 Domain B 的 V_j，参数为 $\text{bsn}\in\{0,1\}^*\cup\{\perp\}$，计数器值为 cnt。$H_i$ 将 m、bsn、cnt 发送给 T。首先 T 验证计数器值为 cnt 的 H_i/M_i 是否是 Passport 群体和 Visa 群体的成员，如果不是，T 将拒绝 H_i/M_i 的请求；否则，T 将 m 和 cnt 交给相应的 M_i，询问是否同意签名。如果 M_i 同意，T 询问 H_i 是否需要签名。如果 H_i 没有退出，T 执行以下步骤。

① 如果 M_i 是假冒的，T 通知 V_j：假冒 TPM 对 m 进行了签名。

② 如果 $\text{bsn}=\perp$，T 通知 V_j：H_i/M_i 已经对 m 进行了签名。

③ 如果 $\text{bsn}\neq\perp$，T 检查 H_i/M_i 是否已经用参数 bsn 和 cnt 对消息进行了签名。如果是，T 在它的跨域假名数据库中查找对应的假名 P，如果不是，将随机生成一个假名 $P\in_R\{0,1\}^{\ell_\sigma}$，$T$ 通知 V_j 假名为 P 的平台对消息 m 签名。

该理想系统模型具有以下安全特性。

(1) 不可伪造性，可信计算平台要对消息 m 做跨域签名，必须满足条件：TPM 的参与；平台本身必须是 Passport 和 Visa 群体的成员；TPM 不是假冒的。

(2) 匿名性，跨域签名都是通过假名 P 进行的。

16.4.3 跨域 DAA 协议

1. IDAA-IssuePassport 协议

该协议是在执行完 DAA-Join 协议，也就是可信计算平台得到 DAA 颁发者(DAA-

Issuer)的签名之后执行的。如果可信计算平台希望与其他域内的验证者交互,先要生成域内的护照证书(Passport Certificate),该护照证书可以与多个不同的签证证书一起使用。

设护照颁发者(Passport Issuer)的公钥为

$$PK_{PI}=(n_{PI},g'_{PI},g_{PI},h_{PI},S_{PI},Z_{PI},R_{PI0},R_{PI1},R_{PI2},R_{PI3},R_{PI4})$$

BCC 方案中的 DAA 颁发者的公钥为:$PK=(n,g',g,h,S,Z,R_0,R_1)$。

主机(Host)首先计算$\varsigma_{PI}=(H_\Gamma(1\parallel \mathrm{bsn}_{PI}))^{(\Gamma-1)/\rho}\bmod \Gamma$,其中 bsn_{PI}是护照颁发者的长期名(long-term base name)。

协议中 TPM 的输入是$(n_{PI},R_{PI0},R_{PI1},S_{PI},\rho,\Gamma)$、$(n,R_0,R_1,S,\rho,\Gamma)$,Host 的输入是$(n_{PI},g'_{PI},g_{PI},h_{PI},S_{PI},Z_{PI},R_{PI0},R_{PI1},R_{PI2},R_{PI3},R_{PI4},n,g',g,h,S,Z,R_0,R_1,\gamma,\rho,\Gamma)$。发放护照证书的具体执行过程如下。

(1) TPM 选择 $v'_{PI}\in_R\{0,1\}^{\ell_n+\ell_\phi}$,使用秘密消息 f_0、f_1 计算 $U_{PI}=R_{PI0}^{f_0}R_{PI1}^{f_1}S_{PI}^{v'_{PI}}\bmod n_{PI}$,$N_{PI}=\varsigma_{PI}^{f_0+f_1 2^{\ell_f}}\bmod \Gamma$,Host 选择 $w,r\in_R\{0,1\}^{\ell_n+\ell_\phi}$,并计算

$$U_{Host}=U_{PI}R_{PI2}^{Ident}\bmod n_{PI},\quad T_1=Ah^w\bmod n,\quad T_2=g^w h^e(g')^r\bmod n$$

其中,$Ident=H(EKPub\parallel Platform\text{-}name\parallel random)\bmod \rho$,random 是随机选择的数,Host 存储 Ident 的值。将 U_{Host}和 N_{PI}发送给护照颁发者。

(2) 执行下面的零知识证明协议:

$$\begin{aligned}&SPK\{(f_0,f_1,v,e,w,r,Ident,v'_{PI}):Z\equiv T_1^{-e}R_0^{f_0}R_1^{f_1}S^v h^{-ew}(\bmod n)\wedge\\&T_2\equiv g^w h^e(g')^r(\bmod n)\wedge 1\equiv T_2^{-e}g^{ew}h^{ee}(g')^{er}(\bmod n)\wedge\\&N_{PI}\equiv\varsigma_{PI}^{f_0+f_1 2^{\ell_f}}(\bmod \Gamma)\wedge U_{Host}\equiv R_{PI0}^{f_0}R_{PI1}^{f_1}S_{PI}^{v_{PI}}R_{PI2}^{Ident}(\bmod n)\wedge\\&f_0,f_1\in\{0,1\}^{\ell_f+\ell_\phi+\ell_H+2}\wedge(e-2^{\ell_e})\in\{0,1\}^{\ell'_e+\ell_\phi+\ell_H+1}\wedge\\&v'_{PI}\in\{0,1\}^{\ell_n+\ell_\phi+\ell_H+2}\}(n_i\parallel n_t)\end{aligned}$$

通过执行该零知识证明协议,可以向护照颁发者证明。

① TPM 拥有 DAA 颁发者的签名。

② U_{Host}中的 f_0、f_1 与 DAA 签名中的 f_0、f_1 是一致的。

具体的协议过程如下。

① TPM 随机选择整数

$$r_{f_0},r_{f_1}\in_R\{0,1\}^{\ell_f+\ell_\phi+\ell_H},\quad r_{v'_{PI}}\in_R\{0,1\}^{\ell_n+2\ell_\phi+\ell_H},\quad r_v\in_R\{0,1\}^{\ell_n+2\ell_\phi+\ell_H}$$

并计算

$$\tilde{U}_{PI}=R_{PI0}^{r_{f_0}}R_{PI1}^{r_{f_1}}S_{PI}^{r_{v'_{PI}}}\bmod n_{PI},\quad \tilde{r}_f=r_{f_0}+r_{f_1}2^{\ell_f}\bmod \rho$$

$$\tilde{N}_{PI}=\varsigma_{PI}^{\tilde{r}_f}\bmod \Gamma,\quad \tilde{T}_{1t}=R_0^{r_{f_0}}R_1^{r_{f_1}}S^{r_v}\bmod n$$

将 $\tilde{T}_{1t}$、$\tilde{N}_{PI}$、$\tilde{U}_{PI}$发送给 Host。

② Host 随机选择

$$\begin{aligned}&r_{Ident}\in_R\{0,1\}^{\ell_f+\ell_\phi+\ell_H},\quad r_e\in_R\{0,1\}^{\ell'_e+\ell_\phi+\ell_H},\quad r_{ee}\in_R\{0,1\}^{2\ell_e+\ell_\phi+\ell_H+1},\\&r_w,r_r\in_R\{0,1\}^{\ell_n+2\ell_\phi+\ell_H},\quad r_{ew},r_{er}\in_R\{0,1\}^{\ell_e+\ell_n+2\ell_\phi+\ell_H+1}\end{aligned}$$

并计算

$$\tilde{T}_1=\tilde{T}_{1t}T_1^{r_e}h^{-r_{ew}}\bmod n,\quad \tilde{T}_2=g^{r_w}h^{r_e}(g')^{r_r}\bmod n,$$

$$\tilde{T}'_2=T_2^{-r_e}g^{r_{ew}}h^{r_{ee}}(g')^{r_{er}}\bmod n,\quad \tilde{U}_{Host}=\tilde{U}_{PI}R_{PI2}^{r_{Ident}}\bmod n_{PI}$$

③ 护照颁发者选择一个随机数 $n_i \in \{0,1\}^{\ell_H}$ 发送给 Host。

④ Host 计算

$$c_h = H(n_{PI} \| R_{PI0} \| R_{PI1} \| R_{PI2} \| S_{PI} \| U_{Host} \| N_{PI} \| \widetilde{U}_{Host} \| \widetilde{N}_{PI} \| n_i \| n \| g \| \\ g' \| h \| R_0 \| R_1 \| S \| Z \| \gamma \| \Gamma \| \rho \| \varsigma_{PI} \| (T_1 \| T_2) \| (\widetilde{T}_1 \| \widetilde{T}_2 \| \widetilde{T}'_2))$$

将 c_h 发送给 TPM，TPM 随机选择 $n_t \in \{0,1\}^{\ell_\Phi}$，计算 $c = H(c_h \| n_t)$。

⑤ TPM 计算 $s_{f_0} = r_{f_0} + c \cdot f_0, s_{f_1} = r_{f_1} + c \cdot f_1, s_{v'_{PI}} = r_{v'_{PI}} + c \cdot v'_{PI}, s_v = r_v + c \cdot v$，并发送 $(c, n_t, s_v, s_{v'_{PI}}, s_{f_0}, s_{f_1})$ 给 Host。Host 计算

$$s_{Ident} = r_{Ident} + c \cdot Ident, \quad s_e = r_e + c \cdot (e - 2^{\ell_e - 1}), \quad s_{ee} = r_{ee} + c \cdot (e^2),$$

$$s_w = r_w + cw, \quad s_{ew} = r_{ew} + c \cdot w \cdot e, \quad s_r = r_r + c \cdot r, \quad s_{er} = r_{er} + c \cdot e \cdot r$$

将以下的信息发送给护照颁发者：

$$(c, n_t, s_v, s_{v'_{PI}}, s_{f_0}, s_{f_1}, s_{Ident}, \varsigma_{PI}, T_1, T_2, s_e, s_{ee}, s_w, s_{ew}, s_r, s_{er})$$

⑥ 护照颁发者计算以下的信息：

$$\hat{U} = U_{Host}^{-c} R_{PI0}^{s_{f_0}} R_{PI1}^{s_{f_1}} S_{PI2}^{s_{Ident}} S^{s_{v'PI}} \bmod n_{PI}, \quad \hat{N}_{PI} = N_{PI}^{-c} \varsigma_{PI}^{s_{f_0} + s_{f_1} 2^{\ell_f}} \bmod \Gamma$$

$$\hat{T}_1 = Z^{-c} T_1^{s_e + c2^{\ell_e - 1}} R_0^{s_{f_0}} R_1^{s_{f_1}} S^{S_v} h^{-s_{ew}} \bmod n, \quad \hat{T}'_2 = T_2^{-c} h^{s_e + c2^{\ell_e - 1}} g^{s_w} (g')^{s_r} \bmod n$$

$$\hat{T}'_2 = T_2^{-(s_e + c2^{\ell_e - 1})} g^{s_{ew}} (g')^{s_{er}} h^{s_{ee}} \bmod n$$

验证

$$c \stackrel{?}{=} H(H(n_{PI} \| R_{PI0} \| R_{PI1} \| R_{PI2} \| S_{PI} \| U_{Host} \| N_{PI} \| \hat{U}_{Host} \| \hat{N}_{PI} \| n_i \| n \| g \| \\ g' \| h \| R_0 \| R_1 \| S \| Z \| \gamma \| \Gamma \| \rho \| \varsigma_{PI} \| (T_1 \| T_2) \| (\hat{T}_1 \| \hat{T}_2 \| \hat{T}'_2))) \| n_t)$$

(3) 护照颁发者检查 $\varsigma_{PI} \stackrel{?}{\equiv} (H_\Gamma(1 \| bsn_{PI}))^{(\Gamma-1)/\rho} (\bmod \Gamma)$，根据已知假冒的 f_0、f_1，检查 $N_{PI} \stackrel{?}{\equiv} \varsigma_{PI}^{f_0 + f_1 2^{\ell_f}} (\bmod \Gamma)$。

(4) 护照颁发者进行签名操作，颁发匿名护照证书，在证书中编码以下信息：①有效日期编码 PvalidDate $\in \{0,1\}^{\ell_f}$；②颁发者标识符编码 PIName $\in \{0,1\}^{\ell_f}$，选择 $v''_{PI} \in [2^{\ell_v - 1}, 2^{\ell_v}]$，$e_{PI} \in [2^{\ell_e - 1}, 2^{\ell_e - 1} + 2^{\ell'_e - 1}]$，计算 $A_{PI} = (Z_{PI} / (U_{Host} R_3^{PvalidDate} R_4^{PIName} S_{PI}^{v''_{PI}}))^{1/e_{PI}}$，发送 $(A_{PI}, e_{PI}, v''_{PI}, PvalidDate, PIName)$ 给 Host。

(5) Host 将 v''_{PI} 发送给 TPM，TPM 计算 $v_{PI} = v'_{PI} + v''_{PI}$，TPM 存储 v_{PI}。Host 存储 $(A_{PI}, e_{PI}, Ident)$，得到的 (A_{PI}, e_{PI}, v_{PI}) 是对 $(f_0, f_1, Ident, PvalidDate, PIName)$ 的 CL 签名。把 $(f_0, f_1, Ident, PvalidDate, PIName, A_{PI}, e_{PI}, v_{PI})$ 称为护照证书。

2. IDAA-IssueVisa 协议

该协议是一个域间协议，是签证颁发者和可信计算平台之间执行的一个协议，通过该协议来颁发签证证书。

设签证颁发者的公钥是

$$PK_{VA} = (n_{VA}, g'_{VA}, g_{VA}, h_{VA}, S_{VA}, Z_{VA}, R_{VA0}, R_{VA1}, R_{VA2}, R_{VA3}, R_{VA4}, R_{VA5})$$

$\varsigma_{VA} = (H_\Gamma(1 \| bsn_{VA}))^{(\Gamma-1)/\rho} \bmod \Gamma$，其中 bsn_{VA} 是签证颁发者的长期名（Long-term base name）。协议中 TPM 的输入是 $(n_{PI}, R_{PI0}, R_{PI1}, S, \rho, \Gamma)$，Host 的输入是 $(n_{VA}, g'_{VA}, g_{VA}, h_{VA}, S_{VA}, Z_{VA}, R_{VA0}, R_{VA1}, R_{VA2}, R_{VA3}, R_{VA4}, R_{VA5}, \gamma, \rho, \Gamma)$。协议执行过程如下。

(1) Host 选择随机数 $k_0, k_1, v'_{VA} \in \{0,1\}^{\ell_H}$，随机选择 $w_{PI}, r_{PI} \in \{0,1\}^{\ell_n+\ell_\phi}$，并计算 $U_{VA} = R_{VA0}^{k_0} R_{VA1}^{k_1} S_{VA}^{v'_{VA}} R_{VA2}^{\text{Ident}} \bmod n_{VA}$，$T_{PI1} = A_{PI} h_{PI}^{w_{PI}} \bmod n_{PI}$，$T_{PI2} = g_{PI}^{w_{PI}} h_{PI}^{e_{PI}} (g'_{PI})^{r_{PI}} \bmod n_{PI}$；TPM 计算 $N_{VA} = \varsigma_{VA}^{f_0 + f_1 2^{\ell_f}} \bmod \Gamma$。

(2) 发送 U_{VA}，T_{PI1}，T_{PI2}，N_{VA}，PValidDate，PIName 给护照颁发者。

(3) 执行下面的零知识证明协议

$$\begin{aligned} \text{SPK}\{(k_0, k_1, v_{PI}, e_{PI}, w_{PI}, r_{PI}, \text{Ident}, v'_{VA}, f_0, f_1):\\ Z_{PI} &\equiv T_{PI1}^{-e_{PI}} R_{PI0}^{f_0} R_{PI1}^{f_1} R_{PI2}^{\text{Ident}} R_{PI3}^{\text{PvalidDate}} R_{PI4}^{\text{PIName}} S_{PI}^{v_{PI}} h_{PI}^{-e_{PI} w_{PI}} \pmod{n_{PI}} \ \land\ T_{PI2} \\ &\equiv g_{PI}^{w_{PI}} h_{PI}^{e_{PI}} (g'_{PI})^{r_{PI}} \pmod{n_{PI}} \ \land\ 1 \\ &\equiv T_{PI2}^{-e_{PI}} g_{PI}^{e_{PI} w_{PI}} h_{PI}^{e_{PI} e_{PI}} (g'_{PI})^{e_{PI} r_{PI}} \pmod{n_{PI}} \ \land\ N_{VA} \\ &\equiv \varsigma_{VA}^{f_0 + f_1 2^{\ell_f}} \pmod{\Gamma} \ \land\ U_{VA} \\ &\equiv R_{VA0}^{k_0} R_{VA1}^{k_1} S_{VA}^{v'_{VA}} R_{VA2}^{\text{Ident}} \pmod{n_{VA}} \ \land\ f_0, f_1 \in \{0,1\}^{\ell_f + \ell_\phi + \ell_H + 2}\} \end{aligned}$$

该零知识证明协议的过程与 IDAA-IssuePassport 协议的第(2)步类似，在此不再赘述，执行该零知识证明协议可以向签证颁发者证明。

① 可信计算平台拥有护照颁发者签名的证书。

② U_{VA} 中 Ident 和护照颁发者颁发的证书中的 Ident 是一致的。

(4) 签证颁发者检查 $\varsigma_{VA} \stackrel{?}{=} (H_\Gamma(1 \| \text{bsn}_{VA}))^{(\Gamma-1)/\rho} \bmod \Gamma$ 和 $N_{VA} \stackrel{?}{\equiv} (\varsigma_{VA}^{f_0 + f_1 2^{\ell_f}}) \pmod{\Gamma}$。

(5) 签证颁发者颁发签证证书给申请者，在证书中编码

① 签证有效期限编码 VvalidDate $\in \{0,1\}^{\ell_f}$。

② 签证颁发者编码 VAName $\in \{0,1\}^{\ell_f}$。

③ 用途、签证的等级编码 level $\in \{0,1\}^{\ell_f}$，该属性说明了签证的用途，如是商务签证、个人签证等。

计算 $A_{VA} = (Z_{VA} / (U R_{VA3}^{\text{VvalidDate}} R_{VA4}^{\text{VAName}} R_{VA5}^{\text{level}} S_{VA}^{v''_{VA}}))^{1/e_{VA}}$。

发送(A_{VA}，e_{VA}，v''_{VA}，VvalidDate，level，VAName)给 Host。

(6) Host 计算 $v_{VA} = v'_{VA} + v''_{VA}$，得到的($A_{VA}$，$e_{VA}$，$v_{VA}$)是对($k_0$，$k_1$，Ident，VvalidDate，VAName，level)的知识签名。把(k_0，k_1，Ident，VvalidDate，VAName，level，A_{VA}，e_{VA}，v_{VA})称为签证证书。

3. IDAA-Sign 操作

IDAA-Sign 是一个签名操作。只有通过 IDAA-Sign 签名操作，可信计算平台才能向不同域的验证者匿名地证明自己的身份。具体步骤如下。

(1) TPM 计算 $N_V = \varsigma_V^{f_0 + f_1 2^{\ell_f}} \bmod \Gamma$，发送 N_V 给 Host。其中 ς_V 和验证者相关。

(2) Host 随机选择 $w_{PI}, r_{PI} \in \{0,1\}^{\ell_n+\ell_\phi}$，计算

$$T_{PI1} = A_{PI} h_{PI}^{w_{PI}} \bmod n_{PI},\quad T_{PI2} = g_{PI}^{w_{PI}} h_{PI}^{e_{PI}} (g'_{PI})^{r_{PI}} \bmod n_{PI}$$

(3) Host 随机选择 $w_{VA}, r_{VA} \in \{0,1\}^{\ell_n+\ell_\phi}$，计算

$$T_{VA1} = A_{VA} h_{VA}^{w_{VA}} \bmod n_{VA},\quad T_{VA2} = g_{VA}^{w_{VA}} h_{VA}^{e_{VA}} (g'_{VA})^{r_{VA}} \bmod n_{VA}$$

(4) 进行知识签名。

$$\text{SPK}\{(f_0, f_1, v_{PI}, e_{PI}, w_{PI}, r_{PI}, \text{Ident}, k_0, k_1, v_{VA}, e_{VA}, w_{VA}, r_{VA}):$$

$$
\begin{aligned}
T_{\mathrm{PI2}} &\equiv g_{\mathrm{PI}}^{w_{\mathrm{PI}}} h_{\mathrm{PI}}^{e_{\mathrm{PI}}} (g'_{\mathrm{PI}})^{r_{\mathrm{PI}}} \pmod{n_{\mathrm{PI}}} \wedge Z_{\mathrm{PI}} \\
&\equiv T_{\mathrm{PI1}}^{e_{\mathrm{PI}}} R_{\mathrm{PI0}}^{f_0} R_{\mathrm{PI1}}^{f_1} R_{\mathrm{PI2}}^{\mathrm{Ident}} R_{\mathrm{PI3}}^{\mathrm{PvalidDate}} R_{\mathrm{PI4}}^{\mathrm{PIName}} S_{\mathrm{PI}}^{v_{\mathrm{PI}}} h_{\mathrm{PI}}^{-e_{\mathrm{PI}} w_{\mathrm{PI}}} \pmod{n_{\mathrm{PI}}} \wedge 1 \\
&\equiv T_{\mathrm{PI2}}^{-e_{\mathrm{PI}}} g_{\mathrm{PI}}^{e_{\mathrm{PI}} w_{\mathrm{PI}}} h_{\mathrm{PI}}^{e_{\mathrm{PI}} e_{\mathrm{PI}}} (g'_{\mathrm{PI}})^{e_{\mathrm{PI}} r_{\mathrm{PI}}} \pmod{n_{\mathrm{PI}}} \wedge T_{\mathrm{VA2}} \\
&\equiv g_{\mathrm{VA}}^{w_{\mathrm{VA}}} h_{\mathrm{VA}}^{e_{\mathrm{VA}}} (g'_{\mathrm{VA}})^{r_{\mathrm{VA}}} \pmod{n_{\mathrm{VA}}} \wedge (Z_{\mathrm{VA}} / (R_{\mathrm{VA0}}^{k_0} R_{\mathrm{VA1}}^{k_1} R_{\mathrm{VA3}}^{\mathrm{VvalidDate}})) \\
&\equiv T_{\mathrm{VA1}}^{e_{\mathrm{VA}}} R_{\mathrm{VA2}}^{\mathrm{Ident}} R_{\mathrm{VA4}}^{\mathrm{VAName}} R_{\mathrm{VA5}}^{\mathrm{level}} S_{\mathrm{VA}}^{v_{\mathrm{VA}}} h_{\mathrm{VA}}^{-e_{\mathrm{VA}} w_{\mathrm{VA}}} \pmod{n_{\mathrm{VA}}} \wedge 1 \\
&\equiv T_{\mathrm{VA2}}^{-e_{\mathrm{VA}}} g_{\mathrm{VA}}^{e_{\mathrm{VA}} w_{\mathrm{VA}}} h_{\mathrm{VA}}^{e_{\mathrm{VA}} e_{\mathrm{VA}}} (g'_{\mathrm{VA}})^{e_{\mathrm{VA}} r_{\mathrm{VA}}} \pmod{n_{\mathrm{VA}}} \wedge \\
&\quad f_0, f_1, \mathrm{Ident}, k_1, k_2 \in \{0,1\}^{\ell_f + \ell_\phi + \ell_H + 2}\}(n_t \parallel m)
\end{aligned}
$$

具体的知识签名过程不再赘述,可以理解为并行地执行了两个 DAA-Sign 操作,最后得到的签名是

$$
\begin{aligned}
\sigma = (&\varsigma_V, (T_{\mathrm{PI1}}, T_{\mathrm{PI2}}), (T_{\mathrm{VA1}}, T_{\mathrm{VA2}}), (\mathrm{PvalidDate}, \mathrm{PIName}), \\
&(\mathrm{VvalidDate}, \mathrm{VAName}, \mathrm{level}), N_V, c, n_t, \\
&(s_{v_{\mathrm{PI}}}, s_{\mathrm{Ident}}, s_{f_0}, s_{f_1}, s_{e_{\mathrm{PI}}}, s_{e_{\mathrm{PI}} e_{\mathrm{PI}}}, s_{w_{\mathrm{PI}}}, s_{e_{\mathrm{PI}} w_{\mathrm{PI}}}, s_{r_{\mathrm{PI}}}, s_{e_{\mathrm{PI}} r_{\mathrm{PI}}}, \\
&s_{k_0}, s_{k_1}, s_{v_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}}, s_{e_{\mathrm{VA}} e_{\mathrm{VA}}}, s_{w_{\mathrm{VA}}}, s_{e_{\mathrm{VA}} w_{\mathrm{VA}}}, s_{r_{\mathrm{VA}}}, s_{e_{\mathrm{VA}} r_{\mathrm{VA}}}))
\end{aligned}
$$

该知识签名能够说明以下几点。

(1) Host/TPM 拥有护照颁发者颁发的护照证书,并且签名时需要 TPM 的参与。

(2) 主机(Host)拥有签证颁发者颁发的签证证书。

(3) 这两个证书绑定的是同一个可信计算平台(通过 Ident 参数绑定)。

4. IDAA-Vefify 操作

IDAA-Vefify 是一个验证操作。验证的公钥为

$(n_{\mathrm{PI}}, g_{\mathrm{PI}}, g'_{\mathrm{PI}}, h_{\mathrm{PI}}, R_{\mathrm{PI0}}, R_{\mathrm{PI1}}, R_{\mathrm{PI2}}, R_{\mathrm{PI3}}, R_{\mathrm{PI4}}, S_{\mathrm{PI}}, Z_{\mathrm{PI}}, \gamma, \Gamma, \rho)$和$(n_{\mathrm{VA}}, g_{\mathrm{VA}}, g'_{\mathrm{VA}}, h_{\mathrm{VA}}, R_{\mathrm{VA0}}, R_{\mathrm{VA1}}, R_{\mathrm{VA2}}, R_{\mathrm{VA3}}, R_{\mathrm{VA4}}, R_{\mathrm{VA5}}, S_{\mathrm{VA}}, Z_{\mathrm{VA}}, \gamma, \Gamma, \rho)$。

具体步骤如下。

(1) 验证者验证(PvalidDate,PIName),(VvalidDate,VAName,level)参数合法性。

(2) 验证者计算

$$
\begin{aligned}
&\hat{T}_{\mathrm{PI1}} = Z_{\mathrm{PI}}^{-c} T_{\mathrm{PI1}}^{s_{e_{\mathrm{PI}}} + c2^{\ell_e - 1}} R_{\mathrm{PI0}}^{s_{f_0}} R_{\mathrm{PI1}}^{s_{f_1}} R_{\mathrm{PI2}}^{s_{\mathrm{Ident}}} R_{\mathrm{PI3}}^{\mathrm{PvalidDate}} R_{\mathrm{PI4}}^{\mathrm{PIName}} S_{\mathrm{PI}}^{s_{v_{\mathrm{PI}}}} h_{\mathrm{PI}}^{-s_{e_{\mathrm{PI}} w_{\mathrm{PI}}}} \bmod n_{\mathrm{PI}}, \\
&\hat{T}_{\mathrm{PI2}} = T_{\mathrm{PI2}}^{-c} g_{\mathrm{PI}}^{s_{w_{\mathrm{PI}}}} h_{\mathrm{PI}}^{s_{e_{\mathrm{PI}}} + c2^{\ell_e - 1}} (g'_{\mathrm{PI}})^{s_{r_{\mathrm{PI}}}} \bmod n_{\mathrm{PI}}, \quad \hat{N}_V = N_V^{-c} \varsigma_V^{s_{f_0} + s_{f_1} 2^{\ell_f}} \bmod \Gamma, \\
&\hat{T}'_{\mathrm{PI2}} = T_{\mathrm{PI2}}^{-(s_{e_{\mathrm{PI}}} + c2^{\ell_e - 1})} g_{\mathrm{PI}}^{s_{e_{\mathrm{PI}} w_{\mathrm{PI}}}} h_{\mathrm{PI}}^{s_{e_{\mathrm{PI}} e_{\mathrm{PI}}}} (g'_{\mathrm{PI}})^{s_{e_{\mathrm{PI}} r_{\mathrm{PI}}}} \bmod n_{\mathrm{PI}}, \\
&\hat{T}_{\mathrm{VA2}} = T_{\mathrm{VA2}}^{-c} g_{\mathrm{VA}}^{s_{w_{\mathrm{VA}}}} h_{\mathrm{VA}}^{s_{e_{\mathrm{VA}}} + c2^{\ell_e - 1}} (g'_{\mathrm{VA}})^{s_{r_{\mathrm{VA}}}} \bmod n_{\mathrm{VA}}, \\
&\hat{T}_{\mathrm{VA1}} = Z_{\mathrm{VA}}^{-c} T_{\mathrm{VA1}}^{s_{e_{\mathrm{VA}}} + c2^{\ell_e - 1}} R_{\mathrm{VA0}}^{s_{f_0}} R_{\mathrm{VA1}}^{s_{f_1}} R_{\mathrm{VA2}}^{s_{\mathrm{Ident}}} R_{\mathrm{VA3}}^{\mathrm{VvalidDate}} R_{\mathrm{VA4}}^{\mathrm{VAName}} R_{\mathrm{VA5}}^{\mathrm{level}} S_{\mathrm{VA}}^{s_{v_{\mathrm{VA}}}} h_{\mathrm{VA}}^{-s_{e_{\mathrm{VA}} w_{\mathrm{VA}}}} \bmod n_{\mathrm{VA}}, \\
&\hat{T}'_{\mathrm{VA2}} = T_{\mathrm{VA2}}^{-(s_{e_{\mathrm{VA}}} + c2^{\ell_e - 1})} g_{\mathrm{VA}}^{s_{e_{\mathrm{VA}} w_{\mathrm{VA}}}} h_{\mathrm{VA}}^{s_{e_{\mathrm{VA}} e_{\mathrm{VA}}}} (g'_{\mathrm{VA}})^{s_{e_{\mathrm{VA}} r_{\mathrm{VA}}}} \bmod n_{\mathrm{VA}}
\end{aligned}
$$

(3) 验证

$$
\begin{aligned}
c \stackrel{?}{=} H(H(&n_{\mathrm{PI}} \parallel R_{\mathrm{PI0}} \parallel R_{\mathrm{PI1}} \parallel R_{\mathrm{PI2}} \parallel \mid R_{\mathrm{PI3}} \parallel R_{\mathrm{PI4}} \mid S_{\mathrm{PI}} \parallel Z_{\mathrm{PI}} \parallel n_{\mathrm{VA}} \parallel g_{\mathrm{VA}} \parallel g'_{\mathrm{VA}} \\
&\parallel h_{\mathrm{VA}} \parallel R_{\mathrm{VA0}} \parallel R_{\mathrm{VA1}} \parallel R_{\mathrm{VA2}} \parallel R_{\mathrm{VA3}} \parallel R_{\mathrm{VA4}} \parallel R_{\mathrm{VA5}} \parallel S_{\mathrm{VA}} \parallel Z_{\mathrm{VA}} \parallel \gamma \parallel \Gamma \parallel
\end{aligned}
$$

$\rho \| \varsigma_V \| N_V \| \hat{N}_V \| (T_{PI1} \| T_{PI2}) \| (\hat{T}_{PI1} \| \hat{T}_{PI2} \| \hat{T}'_{PI2}) \| (T_{VA1} \| T_{VA2}) \|$

$(\hat{T}_{VA1} \| \hat{T}_{VA2} \| \hat{T}'_{VA2}))) \| n_t \| m)$

$N_V, \varsigma_V \overset{?}{\in} <\gamma>, s_{f_0}, s_{f_1} \overset{?}{\in} \{0,1\}^{\ell_f+\ell_\phi+\ell_H+1}$,

$s_{e_{PI}}, s_{e_{VA}} \overset{?}{\in} \{0,1\}^{\ell_e'+\ell_\phi+\ell_H+1}, N_V \overset{?}{\equiv} (\varsigma_V^{f_0+f_1 2^{\ell_f}}) (\bmod \Gamma)$

16.4.4 跨域 DAA 协议安全性证明

定理 16.5 在<γ>群的 DDH 假设和强 RSA 假设下，IDAA 方案安全地实现了一个跨域的直接匿名证明系统。

16.4.1 小节给出了跨域 DAA 的理想系统模型，下面将在理想系统中构造模拟器 S。模拟器 S 将在理想系统中代表被攻陷的参与方与 T 交互，并且模拟现实系统中的攻击方 A。模拟器 S 在本地对 A 进行黑盒访问(Black-box access)，获得 A 在协议的真实执行中发送的信息，然后提供给 A 所期望接收到的信息。

文献[32]中给出了 BCC 方案的详细证明过程，由于我们的方案构建在 BCC 方案的基础之上，因此在证明过程中，证明 IDAA 方案的安全性并不困难，为了节省篇幅，文献[32]中已经证明过的内容将不再赘述。本小节将重点介绍 IDAA-Sign/Verify 的模拟，在下面的讨论中，沿用了文献[32]中使用的标记，大写字母表示该参与方没有被攻陷(Corrupted)，小写字母表示已经被攻陷。

1. IDAA-Sign 的模拟

在主机被攻陷而 TPM 没有被攻陷的情况下，被攻陷的主机请求诚实(Honest)的 TPM 对消息进行签名。这时模拟器 S 得到攻击方 A 发送给诚实的 TPM 的消息，要求 TPM M_i 对消息 m 做签名。模拟器 S 将执行如下操作。

(1) 如果带有计数器 cnt_i 的 TPM M_i 没有成功地执行 DAA-Join 操作或者 IDAA-PassportJoin 和 IDAA-VisaJoin 操作，模拟器将拒绝 A 的请求。

(2) ① 模拟器从 A 处得到ς，在记录中查找ς，如果ς存在，将找到对应的 N_V，否则随机选择一个 $N_V \in <\varsigma>$。

② 模拟器 S 发送 N_V 给 A。

(3) S 如下伪造 TPM 的签名操作(代表 TPM 执行下面的操作)。

① S 随机选择整数

$$s_{v_{PI}} \in_R \{0,1\}^{\ell_v+\ell_\phi+\ell_H}, s_{f_0}, s_{f_1} \in_R \{0,1\}^{\ell_f+\ell_\phi+\ell_H}$$

② S 随机选择 $c \in \{0,1\}^{\ell_H}$。

③ S 计算

$$\widetilde{T}_{PI1t} = Z_{PI}^{-c} R_{PI0}^{s_{f_0}} R_{PI1}^{s_{f_1}} S_{PI}^{s_{v_{PI}}} \bmod n_{PI},$$

$\widetilde{N}_V = N_V^{-c} \varsigma^{s_{f_0}+s_{f_1} 2^{\ell_f}} \bmod \Gamma$，发送 $\widetilde{N}_V, \widetilde{T}_{PI1t}$ 给 A。

④ S 从 A 处接收到 c_h，随机选择 n_t，完善 S 的随机预言机($c_h \| n_t \| m$ 与 c 对应)，使得 $c = H(c_h \| n_t \| m)$。

(4) S 在理想系统中控制 H_i，代表 H_i 向 T 请求诚实的 TPM 对 m 签名。同时发送给

A 以下消息：$c, n_t, s_{f_0}, s_{f_1}, s_{v_{PI}}$。

2. IDAA-Verify 的模拟

在主机和 TPM 没有被攻陷(H)的情况下，T 通知 S(作为理想系统中的验证者)：可信计算平台 H_i/M_i 已经使用 bsn 对消息 m 做了签名。在这种情况下，S 需要模拟在现实系统中与 A(作为现实系统中的验证者)交互，伪造跨域签名，S 执行以下操作。

(1) ① 如果 bsn$=\perp$，模拟器 S 随机选择$\varsigma \in \langle \gamma \rangle, N_V \in_R \langle \varsigma \rangle$。

② 如果 bsn$\neq \perp$，S 在跨域假名数据库记录中查找 P，如果 P 存在，查找对应的ς、N_V，如果没有找到，计算$\varsigma=(H_\Gamma(1 \| \text{bsn}))^{(\Gamma-1)/\rho} \bmod \Gamma, N_V \in_R \langle \varsigma \rangle$。

③ S 随机选择 $T_{PI1} \in_R \langle h_{PI} \rangle, T_{PI2} \in_R \langle g'_{PI} \rangle, T_{VA1} \in_R \langle h_{VA} \rangle, T_{VA2} \in_R \langle g'_{VA} \rangle$。

(2) S 如下伪造签名：

① S 计算。

$$s_{v_{PI}} \in_R \{0,1\}^{\ell_v+\ell_\phi+\ell_H}, \quad s_{k_0}, s_{k_1}, s_{f_0}, s_{f_1} \in_R \{0,1\}^{\ell_f+\ell_\phi+\ell_H},$$

$$s_{e_{PI}} \in_R \{0,1\}^{\ell'_e+\ell_\phi+\ell_H}, \quad s_{e_{PI}e_{PI}} \in_R \{0,1\}^{\ell'_e+\ell_\phi+\ell_H+\ell_e+1},$$

$$s_{w_{PI}} \in_R \{0,1\}^{\ell_n+2\ell_\phi+\ell_H}, \quad s_{e_{PI}w_{PI}} \in_R \{0,1\}^{\ell_e+2\ell_\phi+\ell_n+\ell_H+1},$$

$$s_{r_{PI}} \in_R \{0,1\}^{\ell_n+2\ell_\phi+\ell_H}, \quad s_{e_{PI}r_{PI}} \in_R \{0,1\}^{\ell_e+2\ell_\phi+\ell_n+\ell_H+1} s_{e_{VA}} \in_R \{0,1\}^{\ell'_e+\ell_\phi+\ell_H},$$

$$s_{e_{VA}e_{VA}} \in_R \{0,1\}^{\ell'_e+\ell_\phi+\ell_H+\ell_e+1}, \quad s_{w_{VA}} \in_R \{0,1\}^{\ell_n+2\ell_\phi+\ell_H},$$

$$s_{e_{VA}w_{VA}} \in_R \{0,1\}^{\ell_e+2\ell_\phi+\ell_n+\ell_H+1}, \quad s_{r_{VA}} \in_R \{0,1\}^{\ell_n+2\ell_\phi+\ell_H},$$

$$s_{e_{VA}r_{VA}} \in_R \{0,1\}^{\ell_e+2\ell_\phi+\ell_n+\ell_H+1}, \quad s_{\text{Ident}} \in_R \{0,1\}^{\ell'_e+\ell_\phi+\ell_H}$$

② S 随机选择 $c \in \{0,1\}^{\ell_H}$。

③ S 计算。

$$\widetilde{T}_{PI1} = Z_{PI}^{-c} T_{PI1}^{s_{e_{PI}}+c2^{\ell_e-1}} R_{PI0}^{s_{f_0}} R_{PI1}^{s_{f_1}} R_{PI2}^{s_{\text{Ident}}} R_{PI3}^{\text{PvalidDate}} R_{PI4}^{\text{PIName}} S_{PI}^{s_{v_{PI}}} h_{PI}^{-s_{e_{PI}w_{PI}}} \bmod n_{PI}$$

$$\widetilde{T}_{PI2} = T_{PI2}^{-c} g_{PI}^{s_{w_{PI}}} h_{PI}^{s_{e_{PI}}+c2^{\ell_e-1}} (g'_{PI})^{s_{r_{PI}}} \bmod n_{PI},$$

$$\widetilde{T}'_{PI2} = T_{PI2}^{-(s_{e_{PI}}+c2^{\ell_e-1})} g_{PI}^{s_{e_{PI}w_{PI}}} h_{PI}^{s_{e_{PI}e_{PI}}} (g'_{PI})^{s_{e_{PI}r_{PI}}} \bmod n_{PI},$$

$$\widetilde{N}_V = N_V^{-c} \varsigma^{s_{f_0}+s_{f_1}2^{\ell_f}} \bmod \Gamma, \quad \widetilde{T}_{VA2} = T_{VA2}^{-c} g_{VA}^{s_{w_{VA}}} h_{VA}^{s_{e_{VA}}+c2^{\ell_e-1}} (g'_{VA})^{s_{r_{VA}}} \bmod n_{VA},$$

$$\widetilde{T}_{VA1} = Z_{VA}^{-c} T_{VA1}^{s_{e_{VA}}+c2^{\ell_e-1}} R_{VA0}^{s_{k_0}} R_{VA1}^{s_{k_1}} R_{VA2}^{s_{\text{Ident}}} R_{VA3}^{\text{VvalidDate}} R_{VA4}^{\text{VAName}} R_{VA5}^{\text{level}} S_{VA}^{s_{v_{VA}}} h_{VA}^{-s_{e_{VA}w_{VA}}} \bmod n_{VA},$$

$$\widetilde{T}'_{VA2} = T_{VA2}^{-(s_{e_{VA}}+c2^{\ell_e-1})} g_{VA}^{s_{e_{VA}w_{VA}}} h_{VA}^{s_{e_{VA}e_{VA}}} (g'_{VA})^{s_{e_{VA}r_{VA}}} \bmod n_{VA}$$

④ S 随机选择 n_t，完善 S 的随机预言机，使得

$$\begin{aligned} c = H(H(& n_{PI} \| R_{PI0} \| R_{PI1} \| R_{PI2} \| R_{PI3} \| S_{PI} \| Z_{PI} \| n_{VA} \| g_{VA} \| N_V \| \widetilde{N}_V \\ & \| g'_{VA} \| h_{VA} \| R_{VA0} \| R_{VA1} \| R_{VA2} \| R_{VA3} \| R_{VA4} \| S_{VA} \| Z_{VA} \| \gamma \| \\ & \Gamma \| \rho \| \varsigma \| (T_{PI1} \| T_{PI2}) \| (\widetilde{T}_{PI1} \| \widetilde{T}_{PI2} \| \widetilde{T}'_{PI2}) \| (T_{VA1} \| T_{VA2}) \\ & \| (\widetilde{T}_{VA1} \| \widetilde{T}_{VA2} \| \widetilde{T}'_{VA2}))) \| n_t \| m) \end{aligned}$$

⑤ S 将得到的签名σ 发送给 A

$$\sigma = (\varsigma, (\widetilde{T}_{PI1}, \widetilde{T}_{PI2}), (\widetilde{T}_{VA1}, \widetilde{T}_{VA2}), N_V, c, n_t,$$

$$(\mathrm{PvalidDate}, \mathrm{PIName}), (\mathrm{VvalidDate}, \mathrm{VAName}, \mathrm{level}),$$

$$(s_{v_{\mathrm{PI}}}, s_{\mathrm{Ident}}, s_{f_0}, s_{f_1}, s_{e_{\mathrm{PI}}}, s_{e_{\mathrm{PI}}e_{\mathrm{PI}}}, s_{w_{\mathrm{PI}}}, s_{e_{\mathrm{PI}}w_{\mathrm{PI}}}, s_{r_{\mathrm{PI}}}, s_{e_{\mathrm{PI}}r_{\mathrm{PI}}},$$

$$s_{k_0}, s_{k_1}, s_{v_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}e_{\mathrm{VA}}}, s_{w_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}w_{\mathrm{VA}}}, s_{r_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}r_{\mathrm{VA}}}))$$

最后，证明环境 ε 不能区分自己是运行在现实系统中还是理想系统中，也就是证明现实系统和理想系统中的输出参数是计算不可区分的。在模拟的过程中 S 扮演了现实系统中的不同角色，模拟器 S 在模拟过程中选择的参数（输出）有

$$N_V, \widetilde{N}_V, T_{\mathrm{PI1}}, T_{\mathrm{PI2}}, T_{\mathrm{VA1}}, T_{\mathrm{VA2}}, s_{v_{\mathrm{PI}}}, s_{\mathrm{Ident}}, s_{f_0}, s_{f_1}, s_{e_{\mathrm{PI}}}, s_{e_{\mathrm{PI}}e_{\mathrm{PI}}}, s_{w_{\mathrm{PI}}}, s_{e_{\mathrm{PI}}w_{\mathrm{PI}}},$$

$$s_{r_{\mathrm{PI}}}, s_{e_{\mathrm{PI}}r_{\mathrm{PI}}}, s_{k_0}, s_{k_1}, s_{v_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}e_{\mathrm{VA}}}, s_{w_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}w_{\mathrm{VA}}}, s_{r_{\mathrm{VA}}}, s_{e_{\mathrm{VA}}r_{\mathrm{VA}}}$$

必须证明这些参数值的统计分布与在现实系统中具体操作执行时指定的相应参数值的统计分布是计算不可区分的。不难分析可以得到，两者确实是不可区分的。

在跨域 DAA 方案中，跨域签名的长度及计算效率是衡量方案性能的重要指标，我们也分析了该方案的签名长度和各个阶段的计算效率，同时通过实验验证了该方案的有效性，详细分析可参阅文献[41]。分析和实验验证表明，为了在多信任域内实现匿名通信，跨域 DAA 方案相比原始的 BCC 方案，计算开销有所增加，签名长度增加了近 1 倍，但是其中 TPM 的计算开销并没有显著增加，增加的计算开销主要由主机完成。

16.5 子群隐私增强保护方案

在 BCC 方案中，其匿名性机制是建立在基名(Base name)的基础上，BCC 方案为基名的选择提供了两种选项：如果基名是随机选择的，那么任意两次生成的签名都是不可关联的(Unlinkable)；如果基名是由验证方选定的，生成的签名是可以关联的，在某一个给定的时间内，如果基名没有变化，那么对于验证者来说在该时间段内签名是可关联的，可信计算平台就没有隐私性可言。在本节中将这种匿名性机制称为"验证者相关的完全或无匿名性"。无论是 BCC 方案还是 HS 方案，都采用了这种"验证者相关的完全或无匿名性"。针对这种匿名性机制，我们提出了更加灵活的匿名性机制，称为子群隐私增强保护方案。其特点是：子群内的成员签名不仅能证明平台是可信计算平台，还能证明该可信计算平台是属于某个特定的群体，该方案为可信计算平台在小群体如局域网内的认证应用提供了解决方案。

16.5.1 子群隐私增强保护安全模型

子群隐私增强保护方案的应用基本框架如图 16.9 所示。

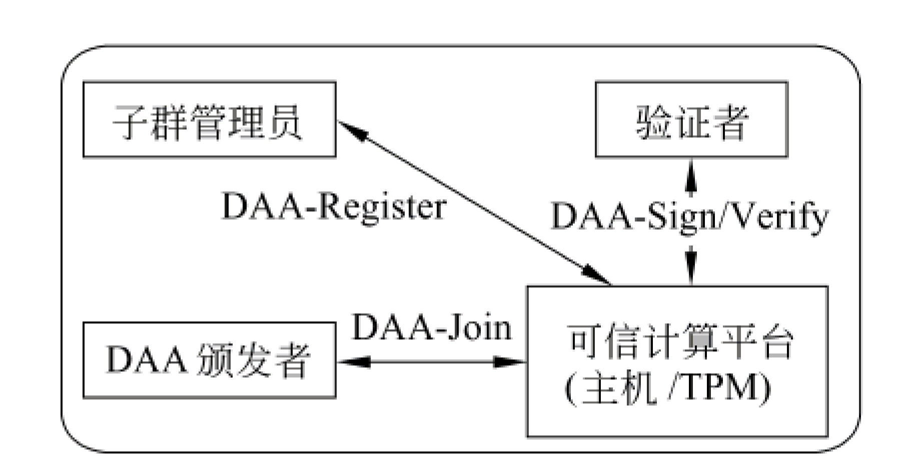


图 16.9 带子群隐私增强保护的直接匿名证明方案框架

子群隐私增强保护方案是在直接匿名证明方案的基础上构建的，除了与 DAA 有相同的参与方外，子群隐私增强保护方案增加了一个参与方：子群管理员。该参与方负责管理子群成员的加入和删除。

定义 16.1 带子群隐私增强保护的直接匿名证明方案(简称 SDAA)是由以下几个参与方构成的签名方案：颁发者(Issuer)、子群管理员、可信计算平台和验证者。它由下面几个过程构成。

(1) 初始化过程(SDAA-Setup)：给定安全参数 1^k，颁发者和子群管理员产生系统公钥和私钥。

(2) 加入过程(SDAA-Join)：是可信计算平台和颁发者之间的一个交互式协议。通过这个协议，可信计算平台得到成员证书和成员私钥。

(3) 注册过程(SDAA-Register)：是可信计算平台与子群管理员之间的一个交互协议，通过这个协议，可信计算平台申请成为子群的一个成员，并发送相关注册信息给子群管理员，子群管理员验证相关信息，将该成员加入相应的子群，并更新子群成员。子群管理员维护两个列表，分别为子群成员有效列表和子群成员撤销列表。

(4) 签名过程(SDAA-Sign)：可信计算平台使用成员证书和成员私钥对给定的消息做匿名签名。

(5) 验证过程(SDAA-Verify)：验证者验证签名的合法性。

(6) 成员更新过程(SDAA-Update)：子群管理员执行的操作，更新子群成员有效列表和子群成员撤销列表。

SDAA 方案必须满足以下安全特性。

(1) 不可伪造性(Unforgeability)：有效的成员证书只有 TPM 和颁发者通过加入过程得到，只有拥有成员证书的可信计算平台才能对消息 m 做匿名签名。

(2) 匿名性(Anonymity)：除非 TPM 出现在撤销列表中，否则不能通过签名确认签名者(即可信计算平台)的身份。可信计算平台的签名不仅对于颁发者是匿名的，对子群管理员也是匿名的。

(3) 不可关联性(Unlinkability)：验证者确定两个不同的签名是否来自于同一个 TPM 在计算上是困难的。

SDAA 方案构建在 BCC 方案的基础上，与 BCC 方案不同的是，增加了一个子群管理员，并且增加了注册过程和成员更新过程，签名过程也与 BCC 方案有所不同，在下面给定的方案描述中，与 BCC 方案相同的过程将不再赘述。

16.5.2 SDAA 方案 Ⅰ

SDAA 方案 Ⅰ 的主要思想是：子群管理员维护一个成员有效列表 E_{add} 和成员撤销列表 E_{del}，首先子群成员通过 SDAA-Register 过程加入子群，在签名时，子群成员 A 做知识签名证明 $\{A: A \in E_{add} \wedge A \notin E_{del}\}$。

SDAA 方案中与 BCC 方案中相同的过程/协议在本节中就不再赘述，在这里主要讨论与 BCC 方案不同的过程/协议。

1. 注册过程(SDAA-Register)

可信计算平台发送基名-假名对 $<\varsigma, N>$ 给子群管理员，其中 N 是通过可信平台模块

的秘密 f 计算出来的($N=\varsigma^f \bmod \Gamma$),子群管理员对可信计算平台进行认证,确认该基名-假名对是由符合条件的可信计算平台产生。子群管理员维护两个列表:基名-假名对有效列表 E_{add},初始时 $E_{add}=\{\}$;基名-假名对撤销列表 E_{del},初始时 $E_{del}=\{\}$。具体步骤如下。

(1) 可信计算平台利用 TPM 持有的秘密消息 f_i 计算 $N_i=\varsigma_i^{f_i} \bmod \Gamma$,并将二元组 $<\varsigma_i,N_i>$ 发送给子群管理员。

(2) 可信计算平台与子群管理员执行零知识证明交互协议 PK$\{f_i:N_i=\varsigma_i^{f_i} \bmod \Gamma\}$,证明 $<\varsigma_i,N_i>$ 是由可信计算平台产生的。

(3) 子群管理员将二元组 $<\varsigma_i,N_i>$ 放入 E_{add} 列表,形成新的基名-假名对有效列表 $E'_{add}=E_{add}\cup\{<\varsigma_i,N_i,P_i>\}=\{<\varsigma_1,N_1,P_1>\cdots<\varsigma_i,N_i,P_i>\cdots<\varsigma_n,N_n,P_n>\}$,其中 P_i 是成员在子群中的标识信息。

2. 签名过程(SDAA-Sign)

首先可信计算平台与验证方协商选取 E_{add} 列表中几个成员作为其匿名隐藏的对象,记为 S_{add},可信计算平台代表 S_{add} 中的成员签名,同时在 E_{del} 中选取集合 S_{del},证明可信计算平台不在集合 S_{del} 中。该签名过程是对 BCC 方案的 DAA-Sign 进行了扩展,分为两个步骤,第一步是进行 BCC 方案的 DAA-Sign 操作,具体的签名可以参见 16.3 节或文献[32];之后进行第二步的证明,具体步骤如下。

(1) 首先从有效列表 E_{add} 中选择 t 个基名-假名对(验证者和签名者协商建立),记为 S_{add},其中 $N_1=\varsigma_1^{f_1},N_2=\varsigma_2^{f_2},\cdots,N_t=\varsigma_t^{f_t}$;并在撤销列表中选择 $S_{del}\subseteq E_{del}$。

(2) 可信计算平台证明 TPM 持有的秘密 f 满足

$$\{f:f\in S_{add}=\{<N_1,\varsigma_1>,\cdots,<N_t,\varsigma_t>\}\wedge$$
$$f\notin S_{del}=\{<N'_1,\varsigma'_1>,\cdots,<N'_t,\varsigma'_t>\}\},$$

为了叙述方便,签名操作分为两步(两步可以合并一起执行):第一步执行 BCC 方案的 DAA-Sign 操作,得到签名 σ_1,第二步执行下面的知识签名得到 σ_2。

$$\sigma_2=\mathrm{SPK}\{f:(N_1\equiv\varsigma_1^f \bmod \Gamma\vee\cdots\vee N_t\equiv\varsigma_t^f \bmod \Gamma)\wedge$$
$$N'_1\neq(\varsigma'_1)^f \bmod \Gamma\wedge\cdots\wedge N'_t\neq(\varsigma'_t)^f \bmod \Gamma\}(m)$$

不失一般性,假设 $N_i=\varsigma_i^f \bmod \Gamma$,其具体的签名操作步骤如下。

① 证明者选择 $v_1,v_2,\cdots,v_t,w_1,\cdots,w_{i-1},w_{i+1},\cdots,w_t$,计算

$$t_1=N_1^{-w_1}\varsigma_1^{v_1} \bmod \Gamma,\cdots,t_{i-1}=N_{i-1}^{-w_{i-1}}\varsigma_{i-1}^{v_{i-1}} \bmod \Gamma,t_i=\varsigma_i^{v_i} \bmod \Gamma,$$
$$t_{i+1}=N_{i+1}^{-w_{i+1}}\varsigma_{i+1}^{v_{i+1}} \bmod \Gamma,\cdots,t_t=N_t^{-w_t}\varsigma_t^{v_t} \bmod \Gamma;$$

② 令 $r=v_i$,对于 $j=1,\cdots,t$,可信计算平台做以下操作。

a. 可信计算平台选择 $x_j\in_R\{0,1\}^{\ell_\rho}$。

b. 可信计算平台计算承诺值

$$U_j=(\varsigma'_j)^{x_j} \bmod \Gamma,V_j=(N'_j)^{x_j} \bmod \Gamma,W_j=U_j^f \bmod \Gamma$$

c. 可信计算平台选择 $r_j\in_R\{0,1\}^{\ell_\rho}$。

d. 可信计算平台计算

$$\widetilde{U}_j=(\varsigma'_j)^{r_j} \bmod \Gamma,\widetilde{V}_j=(N'_j)^{r_j} \bmod \Gamma,\widetilde{W}_j=U_j^r \bmod \Gamma$$

③ 计算

$$c=H(\varsigma_1\parallel N_1\parallel\cdots\parallel\varsigma_t\parallel N_t\parallel\varsigma'_1\parallel N'_1\parallel\cdots\parallel\varsigma'_t\parallel N'_t\parallel t_1,\cdots,$$

$$\| t_t \| \widetilde{U}_1 \| \widetilde{V}_1 \| \widetilde{W}_1 \| \cdots \| \widetilde{U}_t \| \widetilde{V}_t \| \widetilde{W}_t \| m)$$

④ 令 $c_1=w_1,\cdots c_{i-1}=w_{i-1},c_i=c-(c_1+c_2+\cdots+c_{i-1}+c_{i+1}+\cdots+c_t),c_{i+1}=w_{i+1},\cdots,c_t=w_t$。

⑤ 对于 $j=1,\cdots,t$,可信计算平台计算 $s_j=r_j+cx_j,s=r+cf$。

⑥ $s_1'=v_1,\cdots,s_{i-1}'=v_{i-1},s_i'=v_i+c_if,s_{i+1}'=v_{i+1},\cdots,s_t'=v_t$。

⑦ 得到的签名为

$$\sigma_2=(U_1,V_1,W_1,\cdots,U_t,V_t,W_t,c_1,\cdots,c_t,s_1,\cdots,s_t,s,s_1',\cdots,s_t')$$

(3) 最后得到签名 $\sigma=(\sigma_1,\sigma_2)$。

3. 验证阶段(SDAA-Verify)

首先利用 BCC 方案的 DAA-Verify 过程验证 σ_1 的合法性,σ_2 的验证过程如下。

(1) 计算

$$\hat{t}_1=N_1^{-c_1}\varsigma_1^{s_1'},\cdots,\hat{t}_i=N_i^{-c_i}\varsigma_i^{s_i'},\cdots,\hat{t}_t=N_t^{-c_t}\varsigma_t^{s_t'} \tag{16-15}$$

(2) 对 $j=1,\cdots,t$,验证者验证

① $U_j,V_j,W_j\overset{?}{\in}\pm\{0,1\}^{\ell_\Gamma},s_j\overset{?}{\in}\pm\{0,1\}^{\ell_\Gamma},V_j\overset{?}{\neq}W_j$。

② 验证者计算

$$\hat{U}_j=U_j^{-c}(\varsigma_j')^{s_j}\bmod\Gamma,\hat{V}_j=V_j^{-c}(N_j')^{s_j}\bmod\Gamma,\hat{W}_j=W_j^{-c}U_j^{s}\bmod\Gamma \tag{16-16}$$

检查以下的等式:

$$c_1+c_2+\cdots+c_t\overset{?}{=}H(\varsigma_1\|N_1\|\cdots\|\varsigma_t\|N_t\|\varsigma_1'\|N_1'\|\cdots\|\varsigma_t'\|N_t'\|\hat{t}_1,\cdots,\|\hat{t}_t\|\hat{U}_1\|\hat{V}_1\|\hat{W}_1\|\cdots\|\hat{U}_t\|\hat{V}_t\|\hat{W}_t\|m)$$

4. 成员更新(SDAA-Update)

管理员如需要撤销某个成员$<\varsigma_i,N_i,P_i>$,将其从列表 E_{add} 中删除即 $E_{add}=E_{add}-\{<\varsigma_i,N_i,P_i>\}$,并将其加入 $E_{del}=E_{del}\cup\{<\varsigma_i,N_i,P_i>\}$。

在 SDAA 方案Ⅰ中维护的有效列表 E_{add} 和撤销列表 E_{del} 可以在公网上发布(由某个 CA 签名认证),从而保证其真实性。

引理 16.1 SDAA 方案Ⅰ中的签名所基于的交互式协议是诚实验证者关于知识 f 的统计零知识证明。

证明 关于该协议的零知识证明是比较直观的,下面主要证明该协议是一个知识证明协议。也就是要给出一个关于所证明知识的提取器(Knowledge Extractor)。假设存在一个知识提取器能回绕调用(Rewind)协议中的证明者。证明者发送 $U_j,V_j,W_j,\widetilde{U}_j,\widetilde{V}_j,\widetilde{W}_j$,$j=1,\cdots,t$ 和 $t_1,\cdots,t_t$ 给验证者,其中 $V_j\neq W_j,j=1,\cdots,t$。为了响应挑战值 c,证明者响应 $s_1,\cdots,s_t,s,s_1',\cdots,s_t'$。为了响应挑战 $c'\neq c$,证明者响应 $\tilde{s}_1,\cdots,\tilde{s}_t,\tilde{s},\tilde{s}_1',\cdots,\tilde{s}_t'$。即给定元组

$$(U_1,V_1,W_1,\widetilde{U}_1,\widetilde{V}_1,\widetilde{W}_1,\cdots,U_t,V_t,W_t,\widetilde{U}_t,\widetilde{V}_t,\widetilde{W}_t,t_1,\cdots,t_t,c_1,\cdots,c_t,s_1,\cdots,s_t,s,s_1',\cdots,s_t')$$

$$(U_1,V_1,W_1,\widetilde{U}_1,\widetilde{V}_1,\widetilde{W}_1,\cdots,U_t,V_t,W_t,\widetilde{U}_t,\widetilde{V}_t,\widetilde{W}_t,t_1,\cdots,t_t,c_1',\cdots,c_t',\tilde{s}_1,\cdots,\tilde{s}_t,\tilde{s},\tilde{s}_1',\cdots,\tilde{s}_t')$$

其中 $c_1+c_2+\cdots+c_t=c,c_1'+c_2'+\cdots+c_t'=c'$,提取出知识 f。

由于方程(16-15)和式(16-16)中的等式成立,因此有

$$N_1^{c_1}\varsigma_1^{s_1'}=N_1^{c_1'}\varsigma_1^{\tilde{s}_1'},\cdots,N_i^{c_i}\varsigma_i^{s_i'}=N_{i-1}^{c_i'}\varsigma_{i-1}^{\tilde{s}_i'},\cdots,N_t^{c_t}\varsigma_t^{s_t'}=N_t^{c_t'}\varsigma_t^{\tilde{s}_t'} \tag{16-17}$$

$$U_1^{-c}(\varsigma_1')^{s_1}=U_1^{-c'}(\varsigma_1')^{\tilde{s}_1},V_1^{-c}(N_1')^{s_1}=V_1^{-c'}(N_1')^{\tilde{s}_1},W_1^{-c}U_1^{s}=W_1^{-c'}U_1^{\tilde{s}},\cdots,$$

$$U_t^{-c}(\varsigma_t')^{s_t}=U_t^{-c'}(\varsigma_t')^{\tilde{s}_t},V_t^{-c}(N_t')^{s_t}=V_t^{-c'}(N_t')^{\tilde{s}_t},W_t^{-c}U_t^{s}=W_t^{-c'}U_t^{\tilde{s}} \tag{16-18}$$

假设

$$\Delta c=c-c',\Delta c_1=c_1-c_1',\cdots,\Delta c_t=c_t-c_t',$$
$$\Delta s_1=s_1-\tilde{s}_1,\cdots,\Delta s_j=s_j-\tilde{s}_j,\cdots,$$
$$\Delta s_t=s_t-\tilde{s}_t,\Delta s=s-\tilde{s},\Delta s_1'=s_1'-\tilde{s}_1',\cdots,$$
$$\Delta s_j'=s_j'-\tilde{s}_j',\cdots,\Delta s_t'=s_t'-\tilde{s}_t'$$

考虑等式(16-17)和式(16-18),变换得到

$$N_1^{\Delta c_1}=\varsigma_1^{\Delta s_1'},\cdots,N_i^{\Delta c_i}=\varsigma_i^{\Delta s_i'},\cdots,N_t^{\Delta c_t}=\varsigma_t^{\Delta s_t'},U_j^{\Delta c}=(\varsigma_j')^{\Delta s_j},$$
$$(N_j')^{\Delta s_j}=V_j^{\Delta c},W_j^{\Delta c}=U_j^{\Delta s},\quad j=1,\cdots,t$$

令 $\hat{x}_j=\Delta s_j/\Delta c \bmod \Gamma,j=1,\cdots,t,\hat{f}=\Delta s/\Delta c=\Delta s_i/\Delta c_i \bmod \Gamma$,并且 $\hat{f}=\Delta s_i/\Delta c_i\in\{\Delta s_1'/\Delta c_1,\Delta s_2'/\Delta c_2,\cdots,\Delta s_i'/\Delta c_i,\cdots,\Delta s_t'/\Delta c_t\}$,得到对于 $j=1,\cdots,t$,

$$(\varsigma_j')^{\hat{x}_j}=U_j,\quad N_j^{\hat{x}_j}=V_j,\quad U_j^{\hat{f}}=W_j,(\varsigma_j')^{\hat{f}}=N_j \tag{16-19}$$

根据方程(16-19),得到

$$(\varsigma_j')^{\hat{x}_j\cdot\hat{f}}=W_j,N_j^{\hat{x}_j}=V_j,j=1,\cdots,t,\ \text{因此}\varsigma_j^{\hat{f}}=W_j^{1/\hat{x}_j},N_j=V_j^{1/\hat{x}_j},j=1,\cdots,t$$

因为 $V_j\neq W_j,j=1,\cdots,t$,所以 $V_j^{1/\hat{x}_j}\neq W_j^{1/\hat{x}_j},j=1,\cdots,t$,因此 $N_j\neq(\varsigma_j')^{\hat{f}},j=1,\cdots,t$,也就是说,知识提取器得到 $\hat{f}$ 使得 $(\varsigma_j')^{\hat{f}}\neq N_j,j=1,\cdots,t$,并且 $\hat{f}\in\{<\varsigma_1,N_1>,\cdots,<\varsigma_t,N_t>\}$ 即 $\{\hat{f}:N_1=(\varsigma_1)^{\hat{f}}\vee\cdots\vee N_1=(\varsigma_i)^{\hat{f}}\cdots\vee N_t=(\varsigma_t)^{\hat{f}}\}$。

定理 16.6　在 $<\gamma>$ 群的 DDH 假设和强 RSA 假设下,SDAA 方案Ⅰ实现了一个安全的直接匿名证明系统。

证明　由定理 16.4 可知,BCC 方案安全地实现了一个直接匿名证明系统,亦即满足不可伪造性、匿名性与不可关联性,而 SDAA 方案Ⅰ是在 BCC 方案基础上增加了子群隐私增强保护特性,由引理 16.1 可得,新增加的签名所基于的协议是一个零知识证明协议,在随机预言机模型下签名中并不会泄露信息,因此扩展后的 BCC 方案即方案Ⅰ仍然满足不可伪造性、匿名性与不可关联性。

16.5.3　SDAA 方案Ⅱ

SDAA 方案Ⅱ的主要思想是基于 CL 累加器[42]。当有成员加入子群时,子群管理员累加其加入标记 e_i,成员签名时需要提供 e_i 在累加值中的证据 u_i。子群管理员维护一个成员有效列表 E_{add}。撤销成员时,子群管理员从累加值中删除对应的 e_i。

1. 子群管理员的初始化过程(SDAA-Setup)

(1) 子群管理员随机选择 ℓ_p 比特的素数 p'、q',使得 $p=2p'+1,q=2q'+1$ 为素数,令 $n=pq$,n 的比特位数为 ℓ_n。

(2) 子群管理员选择 u,r,g,h 是群 $QR(n)$ 的生成元。

(3) 子群管理员维护加入成员累加值 u 和撤销成员累加值 r，一个子群成员列表 E_{add} 和一个子群撤销成员列表 E_{del}，列表初始为空。

2. 签名过程(SDAA-Sign)

该签名过程是对 BCC 方案的 DAA-Sign 的扩展，分为两个步骤，第一步是进行 BCC 方案的 DAA-Sign 操作，具体的签名可参见 16.3 节或文献[32]，得到签名 σ_1；之后进行第二步的证明，得到签名 σ_2 的具体步骤如下。

(1) 随机选择 $w_1, w_2, w_3 \in_R \{0,1\}^{2\ell_p}$，计算承诺值 $T_1 = g^{e_i} h^{w_1}$，$T_2 = u_i h^{w_2}$，$T_3 = g^{w_2} h^{w_3}$。

(2) 计算知识签名：$\text{SPK}\{e_i, w_1, w_2, w_3 : T_1 = g^{e_i} h^{w_1} \wedge T_3 = g^{w_2} h^{w_3} \wedge u = T_2^{e_i} h^{-e_i w_2}\}(m)$，计算过程如下。

① 计算辅助值 $\delta_1 = e_i w_2$，$\delta_2 = e_i w_3$，选择

$$r_{e_i}, r_{w_1}, r_{w_2}, r_{w_3}, r_{\delta_1}, r_{\delta_2}$$

计算

$$R_1 = g^{r_{e_i}} h^{r_{w_1}}, \quad R_2 = g^{r_{w_2}} h^{r_{w_3}}, \quad R_3 = T_2^{r_{e_i}} h^{-r_{\delta_1}}, \quad R_4 = T_3^{r_{e_i}} g^{-r_{\delta_1}} h^{-r_{\delta_2}}$$

② 计算 $c = H(g \| h \| T_1 \| T_2 \| T_3 \| R_1 \| R_2 \| R_3 \| R_4 \| m)$。

③ 计算

$$s_{e_i} = r_{e_i} + c e_i, \quad s_{w_1} = r_{w_1} + c w_1, \quad s_{w_2} = r_{w_2} + c w_2$$
$$s_{w_3} = r_{w_3} + c w_3, \quad s_{\delta_1} = r_{\delta_1} + c\delta_1, \quad s_{\delta_2} = r_{\delta_2} + c\delta_2$$

④ 得到的签名 $\sigma_2 = (c, T_1, T_2, T_3, s_{e_i}, s_{w_1}, s_{w_2}, s_{w_3}, s_{\delta_1}, s_{\delta_2})$。

(3) 最后得到签名 $\sigma = (\sigma_1, \sigma_2)$。

3. 验证过程(SDAA-Verify)

首先利用 BCC 方案的 DAA-Verify 过程验证 σ_1 的合法性，σ_2 的验证过程如下。

计算

$$R_1' = g^{s_{e_i}} h^{s_{w_1}} T_1^{-c}, R_2' = g^{s_{w_2}} h^{s_{w_3}} T_3^{-c}, R_3' = T_2^{s_{e_i}} h^{-s_{\delta_1}} u^{-c},$$
$$R_4' = T_3^{s_{e_i}} g^{-s_{\delta_1}} h^{-s_{\delta_2}} \tag{16-20}$$

验证 $c \stackrel{?}{=} H(g \| h \| T_1 \| T_2 \| T_3 \| R_1' \| R_2' \| R_3' \| R_4' \| m)$。

4. 成员更新(SDAA-Update)

(1) 增加新成员 e_i 后子群中的成员更新证据 $u_i = u_i^{e_i}$，子群管理员更新累加值 $u = u^{e_i} \bmod n$，$E_{\text{add}} = E_{\text{add}} \cup \{e_i\}$。

(2) 撤销一个成员时(成员标识为 $\tilde{e}$)，根据扩展欧几里德算法，计算 a、b 使得 $a e_i + b\tilde{e} = 1$，更新成员证据：$u_i = u_i^b u^a$。

(3) $E_{\text{add}} = E_{\text{add}} - \{\tilde{e}\}$，$E_{\text{del}} = E_{\text{del}} \cup \{\tilde{e}\}$。

引理 16.2 在强 RSA 假设下，令 n 是 RSA 模数，给定 $u, g \in \text{QR}(n)$，并且 g 是群 $\text{QR}(n)$ 的生成元，存在 $x, y \in Z_n$，使得 $g^x \equiv u^y \pmod n$，那么 $y|x$。

证明 令 x 和 y 的最大公因子为 r，即 $\text{GCD}(x,y)=r$，那么根据扩展欧几里德算法，可找到 α、β，使得 $\alpha x + \beta y = r$，那么 $g = g^{(\alpha x + \beta y)/r} = (u^{\alpha y} g^{\beta y})^{1/r} = (u^{\alpha} g^{\beta})^{y/r}$，如果 $y > r$，那么根据该式，就可以求出 g 的 y/r 次方根，这与强 RSA 假设矛盾，又因为 $\text{GCD}(x,y)=r$，所以 $y=r$，即 $y|x$。

引理 16.3 在强 RSA 假设下，SDAA 方案 Ⅱ 中的签名所基于的交互式协议是诚实验

证者关于知识 e_i、u_i 的统计零知识证明。

证明　协议的完备性和零知识性容易证明，下面证明其合理性，也就是要给出一个关于所证明知识 e_i、u_i 的提取器(Knowledge extractor)。在交互式协议中，知识提取器回应两个不同的挑战值，得到两组可接受的值$(T_1,T_2,T_3,c,s_{e_i},s_{w_1},s_{w_2},s_{w_3},s_{\delta_1},s_{\delta_2})$和$(T_1,T_2,T_3,\tilde{c},\tilde{s}_{e_i},\tilde{s}_{w_1},\tilde{s}_{w_2},\tilde{s}_{w_3},\tilde{s}_{\delta_1},\tilde{s}_{\delta_2})$，假设

$$\Delta c=\tilde{c}-c,\quad \Delta e_i=\tilde{S}_{e_i}-S_{e_i},\quad \Delta w_1=\tilde{s}_{w_1}-s_{w_1},\quad \Delta w_2=\tilde{s}_{w_2}-s_{w_2},$$

$$\Delta w_3=\tilde{s}_{w_3}-s_{w_3},\quad \Delta\delta_1=\tilde{s}_{\delta_1}-s_{\delta_1},\quad \Delta\delta_2=\tilde{s}_{\delta_2}-s_{\delta_2}$$

由式(16-20)可得

$$R_1'=g^{s_{e_i}}h^{s_{w_1}}T_1^{-c}=g^{\tilde{s}_{e_i}}h^{\tilde{s}_{w_1}}T_1^{-\tilde{c}},$$

$$R_2'=g^{s_{w_2}}h^{s_{w_3}}T_3^{-c}=g^{\tilde{s}_{w_2}}h^{\tilde{s}_{w_3}}T_3^{-\tilde{c}} \tag{16-21}$$

$$R_3'=T_2^{s_{e_i}}h^{-s_{\delta_1}}u^{-c}=T_2^{\tilde{s}_{e_i}}h^{-\tilde{s}_{\delta_1}}u^{-\tilde{c}},$$

$$R_4'=T_3^{s_{e_i}}g^{-s_{\delta_1}}h^{-s_{\delta_2}}=T_3^{\tilde{s}_{e_i}}g^{-\tilde{s}_{\delta_1}}h^{-\tilde{s}_{\delta_2}} \tag{16-22}$$

由式(16-21)和式(16-22)两式可得 $T_1^{\Delta c}=g^{\Delta e_i}h^{\Delta w_1}$，$T_3^{\Delta c}=g^{\Delta w_2}h^{\Delta w_3}$，$T_2^{\Delta e_i}=h^{\Delta\delta_1}u^{\Delta c}$，$T_3^{\Delta e_i}=g^{\Delta\delta_1}h^{\Delta\delta_2}$，由引理 16.5 可得 $\Delta c\,|\,\Delta e_i$，$\Delta c\,|\,\Delta w_1$，因此 $e_i=\dfrac{\Delta e_i}{\Delta c}$。

由于 $T_3^{\Delta c}=g^{\Delta w_2}h^{\Delta w_3}$ 和 $T_3^{\Delta e_i}=g^{\Delta\delta_1}h^{\Delta\delta_2}$，可得到 $g^{\Delta c\Delta\delta_1}h^{\Delta c\Delta\delta_2}=g^{\Delta e_i\Delta w_2}h^{\Delta e_i\Delta w_3}$，$\Delta\delta_1=\dfrac{\Delta e_i\Delta w_2}{\Delta c}$。

$u=T_2^{\Delta e_i/\Delta c}/h^{\Delta\delta_1/\Delta c}$，$u=u_i^{e_i}$，将 $\Delta\delta_1=\dfrac{\Delta e_i\Delta w_2}{\Delta c}$代入，可得到 $u_i=\dfrac{T_2}{h^{\frac{\Delta w_2}{\Delta c}}}$；从而可得到知识 $(e_i,u_i)=\left(\dfrac{\Delta e_i}{\Delta c},\dfrac{T_2}{h^{\frac{\Delta w_2}{\Delta c}}}\right)$。

定理 16.7　在<γ>群的 DDH 假设和强 RSA 假设下，方案Ⅱ实现了一个安全的直接匿名证明系统。

证明　根据定理 16.4 可得，BCC 方案安全地实现了一个直接匿名证明系统，也即满足不可伪造性、匿名性与不可关联性，而 SDAA 方案Ⅱ是在 BCC 方案基础上增加了子群隐私增强保护特性，由引理 16.3 可得，新增加的签名所基于的协议是一个零知识证明协议，在随机预言机模型下签名中并不会泄露信息，因此扩展后的 BCC 方案即 SDAA 方案Ⅱ仍然满足不可伪造性、匿名性与不可关联性。

16.5.4　SDAA 方案Ⅰ和方案Ⅱ比较分析

从实现上来看，两种方案都是 BCC 方案的一种扩展，并且都能在 TPM v1.2 的基础上实现。本小节将比较 SDAA 方案Ⅰ和方案Ⅱ的不同。

上述两个方案都增加了 BCC 方案的灵活性。由于在计算过程中 TPM 的计算量是一个非常重要的性能指标，因此将比较在签名和注册过程中 TPM 的计算量。

SDAA 方案Ⅰ随着子群成员数的增加，签名和验证的效率都会降低。而 SDAA 方案Ⅱ的签名长度及签名效率不会因为子群成员数的增加而增加。签名和验证算法中的主要运算为模幂、模乘，分别用 E、M 表示，这里主要从签名和验证、注册操作方面比较这两个方案(见表 16.2)，表中 t 表示子群成员的个数。SDAA 方案Ⅰ中签名算法和验证算法的计算量与成员个数线性相关，特别是在较大群体中，随着 t 的增大，签名算法和验证算法的效率将

会明显降低。SDAA 方案Ⅱ的验证算法的计算量为常量，在较大群体中效率高于 SDAA 方案Ⅰ。SDAA 方案Ⅰ中签名长度、签名和验证算法的计算量和子群成员个数线性相关，在 t 不是很大的情况下较为实用，SDAA 方案Ⅱ解决了方案Ⅰ的不足，签名长度、签名和验证算法的计算量均独立于子群成员个数。最后，SDAA 方案Ⅰ的一个额外的特性是不仅能证明可信计算平台属于某个特定群体，同时还能证明不属于某个群体，而方案Ⅱ做不到这一点；方案Ⅰ中的操作需要 TPM 的参与，因此其安全性根植于 TPM，而方案Ⅱ中的操作不需要 TPM 的参与，如果攻击方攻破主机，攻击方就可以提取 e_i、u_i，安全性相对较低。

表 16.2　SDAA 方案Ⅰ和方案Ⅱ的性能比较

	签名操作/TPM 计算量	验证操作	注册操作/TPM 计算量	签名长度(B)
方案Ⅰ	$(6t-1)$E+$(2t-1)$M/$2t$E+2M	$8t$E+$4t$M	1E+1M/1E+1M	$4896t-1632$
方案Ⅱ	14E+14M/0E+0M	12E+8M	0	10926

16.6　基于双线性映射的直接匿名证明方案

在相同的安全强度下，椭圆曲线密码体制要比基于大整数分解和有限域上离散对数的密码体制效率高，并且有更短的私钥。椭圆曲线密码体制的这个特性，使得它更适合于构建下一代 TPM，这是因为以下几点原因。

(1) 目前 TPM1.2 中采用的加解密方案是基于 RSA 体制的，相比 ECC 体制，RSA 体制的计算效率比较低，目前很多的密码系统都是基于椭圆曲线来实现的，尤其是中国的 TCM 已经率先采用了 ECC 体制[3]。

(2) 目前的隐私性保护方案 BCC 方案过于复杂，计算量大，难以部署，需要有更有效的替代方案，并且 BCC 方案并不适合于移动计算平台，因此有必要设计一种新的更有效的隐私性保护方案能同时满足移动平台的需要。

直接匿名证明方案是在群签名方案的基础上发展起来的，与群签名方案不同的是在直接匿名证明方案中，管理员不能对成员进行匿名性的撤销，成员签名对管理员也是匿名的，同时直接匿名证明方案的直接应用场景是可信计算平台(包括两部分，即 TPM/TCM 和主机)。并且在直接匿名证明方案中必须提供一种假冒 TPM/TCM 的检测机制，防止假冒 TPM/TCM 的欺骗。

我们在短群签名方案[44,45]的基础上，基于 q-SDH 假设与 DDH 假设提出了一种新的基于双线性映射的直接匿名证明方案[43]即 BM-DAA 方案(简称 CF 方案)，CF 方案是目前提出的方案中效率最高的，并且拥有最短的签名长度。CF 方案与 BCC 方案、HS 方案相比，具有更短的签名长度，签名长度为 2044b，并且在签名过程中计算效率更高。

16.6.1　CF 方案

CF 方案[43]的基本思想是：首先 TPM/TCM 与颁发者执行加入过程。TPM/TCM 选择随机秘密值 f，计算承诺值 $C=g^f h^{t'}$，并且向颁发者零知识证明 TPM/TCM 知道秘密值 f 和 t'，颁发者验证通过后给可信计算平台颁发成员证书(A,x,t'')，主机存储(A,x)，TPM/TCM

存储$(f,t=t'+t'')$，满足 $e(A,Yg_2^x)=e(g_1,g_2)\cdot e(g^f,g_2)\cdot e(h^t,g_2)$。然后在签名阶段，可信计算平台计算知识签名 SPK$\{f,x,t:e(A,Yg_2^x)=e(g_1,g_2)\cdot e(g^f,g_2)\cdot e(h^t,g_2)\}(m)$。下面详细说明各个过程。

1. 密钥生成算法

给定安全参数 1^k，颁发者选择群 $G_1=<g_1>,G_2=<g_2>,G_T=<g_T>,G_3=<g_3>$，存在可计算的双线性映射 $e:G_1\times G_2\to G_T$ 和可计算的同构 $\psi:G_2\to G_1,\psi(g_2)=g_1$，这些群的阶均是长度为 k 的素数 p。颁发者随机选择 $\gamma\in_R Z_p$ 和$(g,h)\in_R(G_1)^2$，计算 $Y=g_2^\gamma$，则颁发者的公、私钥对为

$$(\mathrm{pk},\mathrm{sk})=((p,g_1,g_2,g_3,g_T,Y,g,h),\gamma)$$

2. 加入过程(Join 协议)

(1) 首先执行 Pedersen 承诺方案，TPM/TCM 选择秘密信息 $f\in_R Z_p$，选择随机数 $t'\in_R Z_p$，计算 $C=g^f h^{t'}$，其中 f 是被承诺的秘密值，将 C 发送给颁发者，之后执行零知识证明协议，证明 TPM/TCM 拥有秘密知识 f 和 t'。具体过程如下。

① TPM/TCM 随机选择$(r_f,r_{t'})\in_R(Z_p)^2$，计算 $C'=g^{r_f}h^{r_{t'}}$，并将 C' 发送给颁发者。

② 颁发者随机选择 $c\in_R Z_p$，并将 c 发送给 TPM/TCM。

③ TPM/TCM 计算 $s_f=r_f+cf,s_{t'}=r_{t'}+ct'$，并将 s_f、$s_{t'}$ 发送给颁发者。

④ 颁发者验证 $C'\stackrel{?}{=}C^{-c}g^{s_f}h^{s_{t'}}$。

(2) 颁发者随机选择 $x\in_R Z_p,t''\in_R Z_p$，计算 $A=(g_1Ch^{t''})^{1/(\gamma+x)}$，并将 A、x、t'' 发送给主机。

(3) 主机存储 A、x，并将 t'' 发送给 TPM/TCM。

(4) TPM/TCM 计算 $t=t'+t''$，存储 f 和 t，主机验证等式(16-23)是否成立，主机验证过程中 $e(g^f,g_2)$ 和 $e(h^t,g_2)$ 的值可以向 TPM/TCM 请求而获得

$$e(A,Yg_2^x)=e(g_1,g_2)\cdot e(g^f,g_2)\cdot e(h^t,g_2) \tag{16-23}$$

通过 Join 协议过程，可信计算平台得到成员证书(A,x,t)及秘密信息 f，其中主机存储(A,x)，TPM/TCM 存储(f,t)。从该协议可以看出，TPM/TCM 可以在保持匿名性的同时使用同一个秘密信息 f 多次申请成员证书。

3. 签名过程(Sign)

CF 方案的签名过程如下。

(1) 主机随机选取 $w\in_R Z_p$，计算 $T_1=Ah^w,T_2=g^w h^{-x}$，T_1 和 T_2 分别是对 A 和 x 的承诺，验证式(16-24)和式(16-25)成立

$$e(T_1,Y)/e(g_1,g_2)=e(h,Y)^w e(h,g_2)^{wx+t}e(g,g_2)^f/e(T_1,g_2)^x \tag{16-24}$$

$$T_2=g^w h^{-x},\quad T_2^{-x}g^{wx}h^{-xx}=1 \tag{16-25}$$

(2) 接下来证明可信计算平台拥有知识 f、x、w、t，使得式(16-24)和式(16-25)成立。利用 Fiat-Shamir 启发式算法，将对知识 f、x、w、t 的零知识证明转换为知识签名，计算辅助值 $\delta_1=wx,\delta_2=-xx$，其中 $H:\{0,1\}^*\to Z_p$。

① 首先 TPM/TCM 随机选取 $r_f,r_t\in Z_p$，计算 $\tilde{R}_1$，并将 $\tilde{R}_1$ 发送给主机

$$\tilde{R}_1=e(g,g_2)^{r_f}e(h,g_2)^{r_t}$$

② 主机随机选取 $r_x, r_w, r_{\delta_1}, r_{\delta_2} \in Z_p$，计算

$$R_1 = \widetilde{R}_1 e(h,Y)^{r_w} e(T_1,g_2)^{r_x} e(h,g_2)^{r_{\delta_1}},$$
$$R_2 = g^{r_w} h^{r_x},\quad R_3 = T_2^{r_x} g^{r_{\delta_1}} h^{r_{\delta_2}}$$

③ 主机计算 $c_h = H(g \| h \| g_1 \| g_2 \| g_T \| Y \| T_1 \| T_2 \| R_1 \| R_2 \| R_3)$，并发送 c_h 给 TPM/TCM。

④ TPM/TCM 随机选择 $n_t \in_R Z_p$，计算 $c = H(H(c_h \| n_t) \| m)$。

⑤ 主机计算 $s_x = r_x + c(-x)$，$s_{\delta_1} = r_{\delta_1} + c\delta_1$，$s_w = r_w + cw$，$s_{\delta_2} = r_{\delta_2} + c\delta_2$；TPM/TCM 计算 $s_f = r_f + cf$，$s_t = r_t + c(-t)$。

(3) 主机输出签名 $\sigma = (T_1, T_2, c, n_t, s_f, s_x, s_t, s_w, s_{\delta_1}, s_{\delta_2})$。

4. 验证过程(Verify)

(1) 给定消息 m 的签名 $\sigma = (T_1, T_2, c, n_t, s_f, s_t, s_x, s_w, s_{\delta_1}, s_{\delta_2})$ 和公钥 $(p, g_1, g_2, g_T, Y, g, h)$。

(2) 计算。

$$R'_1 = e(g,g_2)^{s_f} e(h,Y)^{s_w} e(h,g_2)^{s_{\delta_1}+s_t} e(T_1,g_2)^{s_x} (e(T_1,Y)/e(g_1,g_2))^{-c},$$
$$R'_2 = T_2^{-c} g^{s_w} h^{s_x},\ R'_3 = T_2^{s_x} g^{s_{\delta_1}} h^{s_{\delta_2}}$$

(3) 验证下列等式是否成立。

$$c \stackrel{?}{=} H(H(H(g \| h \| g_1 \| g_2 \| g_T \| Y \| T_1 \| T_2 \| R'_1 \| R'_2 \| R'_3) \| n_t) \| m)$$

5. 可变匿名性机制

上面给出的签名对验证方来说是完全匿名的，为了达到可变匿名性，可信计算平台在产生签名时，使用 TPM/TCM 的秘密 f 计算一个承诺值 T_3，同时计算时将选择一个签名唯一标识符(Solely Signature IDentifier，SSID)。可信计算平台在签名时如果选择的 SSID 是相同的，那么由该可信计算平台生成的签名是可关联的(Linkability)，如果可信计算平台在生成签名时，随机选择 SSID，那么生成的签名是完全匿名的。SSID 在选择时可以由 TPM/TCM 和验证者共同协商确定。

为了提供可变的匿名性机制，在执行签名操作时，同时执行下面的运算，下面是群 G_3 中的计算，即

$$\eta = H_1(\text{SSID}),\quad T_3 = \eta^f, R_4 = \eta^{r_f},\quad R'_4 = T_3^{-c}\eta^{s_f}$$
$$c = H(H(H(\eta \| g \| h \| g_1 \| g_2 \| g_3 \| g_T \| Y \| T_1 \| T_2 \| T_3 \| R_1 \| R_2 \| R_3 \| R_4) \| n_t) \| m)$$

其中 $H_1: \{0,1\}^* \to G_3$。加入匿名性机制之后输出的签名是 $\sigma = (\eta, T_1, T_2, T_3, c, n_t, s_f, s_t, s_x, s_w, s_{\delta_1}, s_{\delta_2})$。

验证签名是否成立的等式为

$$c \stackrel{?}{=} H(H(H(\eta \| g \| h \| g_1 \| g_2 \| g_3 \| g_T \| Y \| T_1 \| T_2 \| T_3 \| R'_1 \| R'_2 \| R'_3 \| R'_4) \| n_t) \| m)$$

6. 假冒 TPM/TCM 检测(Rogue Tagging)

如果 TPM/TCM 内部的秘密 f 泄露，验证者在签名验证时必须对 TPM/TCM 进行检测，以确定签名是否来自于被攻陷的 TPM/TCM。检测的方法是：将已经泄露的 TPM/

TCM 秘密信息 f 加入到撤销列表中，撤销列表中保存了所有假冒 TPM/TCM 的秘密 f，对于在撤销列表中的 f，验证者计算：

$$T_3 \stackrel{?}{=} \eta^f$$

如果存在某个 f 使得等式成立，那么该签名来自假冒的或者已经撤销的 TPM/TCM。

16.6.2 CF 方案的安全性证明

定理 16.8 在 q-SDH 假设和群 G_3 的 DDH 假设下，CF 方案安全地实现了一个直接匿名证明系统。

与文献[32]中的安全性证明类似，本节将采用现实系统/理想系统（Real-system/ideal-system）模型[46,47]来证明 CF 方案的安全性。

1. 理想系统可信方 T

下面首先给出 CF 方案理想系统的可信方 T，该可信方与文献[32]中给出的可信方基本上是一致的。在理想系统中有以下的参与方：一个颁发者 I，一个身份为 id_i 可信平台模块 TPM/TCM，记为 M_i，一个带 TPM/TCM 的可信计算平台 H_i，一个验证者 V_j。下面将给出 CF 方案的理想系统中的可信方 T 所支持的操作。

(1) 初始化操作(Setup)：每个参与方与 T 交互，表明该参与方是否已经被攻击方攻陷(Corrupted)。

(2) 加入操作(Join)：可信计算平台 H_i 向 T 发出请求，希望成为群成员，T 询问 M_i 是否希望 H_i 成为群中的一员，如果 M_i 同意，T 向颁发者 I 发送消息表明身份为 id_i 的可信计算平台希望加入，如果 M_i 是假冒的，T 将向颁发者 I 表明这一点。如果 I 批准，T 向 H_i 通知其已经成功地加入。

(3) 签名/验证操作(Sign/Verify)：H_i 拟对消息 m 进行签名，用的签名唯一标识符为 $\mathrm{SSID}\in\{0,1\}^*\cup\{\perp\}$。$H_i$ 将 m、SSID 发送给 T。首先 T 验证 H_i/M_i 是否为群的成员，如果不是，T 将拒绝 H_i/M_i 的请求；否则，T 将 m 交给相应的 M_i，询问是否同意签名。如果 M_i 同意，T 询问 H_i 是否需要签名。如果 H_i 没有退出，T 执行以下步骤。

① 如果 M_i 是假冒的，T 通知 V_j：假冒 TPM/TCM 对 m 进行了签名。

② 如果 $\mathrm{SSID}=\perp$，T 通知 V_j：H_i/M_i 已经对 m 进行了签名。

③ 如果 $\mathrm{SSID}\neq\perp$，T 检查 H_i/M_i 是否已经用参数 SSID 对消息进行了签名。如果是，则 T 在它的假名数据库中查找对应的假名 P；如果不是，将随机生成一个假名 $P\in_R G_3$，T 通知 V_j：假名为 P 的平台对消息 m 进行了签名。

该理想系统具有以下安全特性。

① 不可伪造性(Unforgeability)：不是群成员的用户或者已经被撤销的群成员不能成功地进行签名操作。

② 可变匿名性(Anonymity)：验证者不能标识出签名者的身份，如果 $\mathrm{SSID}=\perp$，签名是完全匿名的，如果 $\mathrm{SSID}\neq\perp$，签名具有部分的匿名性，验证者通过假名 P 标识签名者。

③ 不可关联性(Unlinkability)：如果 $\mathrm{SSID}=\perp$，验证者无法区分两个不同的签名是否由同一个可信计算平台签发。

2. 模拟器 S

下面将在理想系统中构造模拟器 S。模拟器 S 将在理想系统中代表被攻陷的参与方与

T 交互，并且模拟现实系统中的攻击方 A。模拟器 S 在本地对 A 进行黑盒访问(Black-box access)，获得 A 在协议的真实执行中发送的信息，然后提供给攻击方 A 所期望接收到的信息。在下面的讨论中，沿用了文献[32]中使用的标记，大写字母表示该参与方没有被攻陷(Corrupted)，小写字母表示该参与方已经被攻陷，如(Ihm)表示颁发者 I 是诚实的参与方，主机和 TPM/TCM 已经被攻陷。

系统初始化的模拟是针对颁发者进行的，分为以下两种情况。

(1) 如果颁发者被攻陷，那么模拟器 S 从攻击方处接收到颁发者的公钥$(p, g_1, g_2, g_3, g_T, Y, g, h)$。

(2) 如果颁发者是诚实的，那么模拟器 S 运行密钥生成算法得到公钥$(p, g_1, g_2, g_3, g_T, Y, g, h)$和私钥 γ。

在 Join 的模拟过程中，根据颁发者 I、主机 H_i 和 TPM/TCM M_i 是否被攻陷，可以分为 6 种情况，分别为(IHM)、(Ihm)、(IhM)、(iHM)、(ihm)、(ihM)，下面将分情况逐一讨论。

(IHM)、(ihm)：在这两种情况下，所有的操作都是在参与方之间进行的，不需要触发模拟器。

(Ihm)：在这种情况下，颁发者 I 没有被攻陷，可信计算平台(h 和 m)被攻陷，如图 16.10 所示。模拟器 S 从攻击方 A 处得到加入请求，S 与 A 交互，运行 Join 协议，在这个过程中，A 同时将 T_3 发送给 S 用于假冒 TPM/TCM 检测，如果 S 第一次接收到 T_3，那么 S 存储 T_3，同时通知可信方 T 表明可信计算平台 H_i 请求加入，模拟器 S 在与可信方通信的过程中将扮演理想系统中的 M_i 的角色。如果可信方 T 同意 H_i 加入，那么模拟器 S 与攻击方 A 将交互完成 Join 协议，如果 T 不同意，那么模拟器 S 将中止协议。

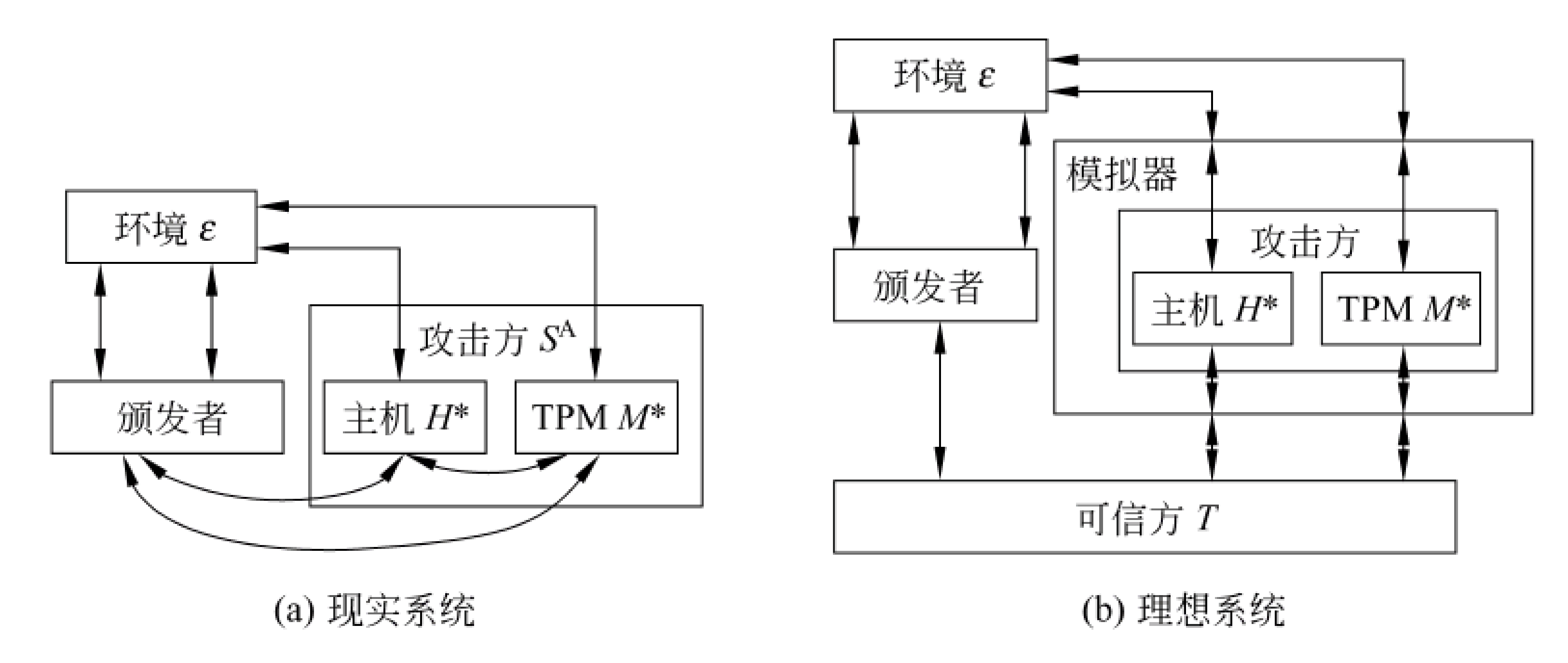


图 16.10 理想系统/现实系统模型(Ihm)

(IhM)：这种情况与(Ihm)类似，与(Ihm)不同的是，模拟器将执行 Join 过程中 TPM/TCM M_i 执行的操作。

(iHM)：在这种情况下，模拟器 S 从可信方 T 处得到平台 id_i 的加入请求。在理想系统中 S 将扮演颁发者的角色，在与攻击方交互的过程中，模拟现实系统中的 H_i/M_i，如果 Join 协议能够成功地完成，那么 S 将从攻击方 A 处得到证书(A, x, t)，S 存储证书及秘密信息 f，并通知 T 允许平台加入，如果 S 没有成功地完成 Join 协议，S 将通知 T 不允许平台加入。

(ihM)：模拟器 S 从攻击方 A 处得到 H_i 的请求，要求 M_i 加入群，模拟器 S 向可信方

T 发送信息表明 H_i 希望加入群。之后，S 从可信方 T 处得到请求消息，表明具有身份 id_i 的平台能否加入，接下来模拟器 S 以 TPM/TCM M_i 运行 Join 协议。如果 Join 协议能够成功地完成，S 从攻击方处得到 t''，存储 t''，通知可信方 T 该平台允许加入；否则，S 通知 T 平台不允许加入。

在 Sign 的模拟过程中，可以分为 3 种情况，分别为(hM)、(hm)、(HM)，下面将分情况逐一讨论。

(HM)、(hm)：在这两种情况中，所有的操作都是在参与方之间进行的，不需要触发模拟器。

(hM)：模拟器 S 从攻击方 A 处得到请求，S 在现实系统中扮演 M_i 角色，模拟器执行以下的操作。

(1) ① 模拟器 S 从 A 处得到 η 和 T_1，S 在记录库中查找，如果 η 存在，模拟器选择对应的 T_3，如果不存在，随机选择 $T_3 \in_R G_3$。

② 发送 T_3 给攻击方 A。

(2) 模拟器 S 伪造签名如下。

① S 随机选择 $s_f, s_t \in_R Z_p$。

② S 随机选择 $c \in_R Z_p$。

③ S 计算

$$\tilde{R}_1 = e(g, g_2)^{s_f} e(h, g_2)^{s_t} (e(T_1, Y)/e(g_1, g_2))^{-c}, \tilde{R}_4 = T_3^{-c} (\eta)^{s_f}$$

并将 $\tilde{R}_1$ 和 $\tilde{R}_4$ 发送给攻击方 A。

④ 模拟器 S 从 A 处得到 c_h，S 随机选择 $n_t \in_R Z_p$，S 完善随机预言机，使得

$$c = H(H(c_h \parallel n_t) \parallel m)$$

(3) S 在理想系统中控制 H_i，代表 H_i 向 T 请求诚实的 TPM/TCM 对 m 签名。同时发送给 A 以下消息：c, n_t, s_f, s_t。

在 Verify 的模拟过程中，可以分为 4 种情况，分别为(HV)、(hv)、(Hv)、(hV)。下面将分情况逐一讨论。

(hv)、(HV)：在这两种情况中，所有的操作都是在参与方之间进行，不需要触发模拟器。

(Hv)：在这种情况下，平台没有被攻陷，模拟器 S 从可信方 T 处得到消息，平台已经对 m 进行了签名。模拟器 S 需要在现实系统中模拟签名。

(1) ① 如果 SSID $=\perp$，模拟器 S 随机选择 $\eta \in_R \langle g_3 \rangle$，$T_3 \in_R \langle g_3 \rangle$。

② 如果 SSID $\neq \perp$，S 在假名数据库记录中查找 P，如果 P 存在，查找对应的 $\langle \eta, T_3 \rangle$，如果没有找到，计算 $\eta = H_1(\mathrm{SSID})$，随机选择 $T_3 \in_R \langle g_3 \rangle$。

③ S 随机选择 $T_1, T_2 \in_R \langle g_1 \rangle$。

(2) S 按以下方式伪造签名。

① S 随机选择 $s_f, s_x, s_w, s_{\delta_1}, s_t, s_{\delta_2} \in_R Z_p$。

② S 随机选择 $c \in_R Z_p$。

③ S 计算

$$R'_1 = e(g, g_2)^{s_f} e(h, Y)^{s_w} e(h, g_2)^{s_{\delta_1}+s_t} e(T_1, g_2)^{s_x} (e(T_1, Y)/e(g_1, g_2))^{-c}$$

$$R'_2 = T_2^{-c} g_1^{s_w} h^{s_x}, \quad R'_3 = T_2^{s_x} g^{s_{\delta_1}} h^{s_{\delta_2}}, \quad R'_4 = T_3^{-c} \eta^{s_f}$$

④ S 选择 n_t，完善随机预言机，使得

$$c \stackrel{?}{=} H(H(H(\eta \| g \| h \| g_1 \| g_2 \| g_3 \| g_T \| Y \| T_1 \| T_2 \| T_3 \| R_1' \| R_2' \| R_3' \| R_4') \| n_t) \| m)$$

(3) S 发送消息 m 的签名 $\sigma=(\eta, T_1, T_2, T_3, c, n_t, s_f, s_t, s_x, s_w, s_{\delta_1}, s_{\delta_2})$ 给攻击方 A。

(hV)：模拟器 S 从攻击方处得到对消息 m 的签名 $\sigma=(\eta, T_1, T_2, T_3, c, n_t, s_f, s_t, s_x, s_w, s_{\delta_1}, s_{\delta_2})$，首先验证签名 σ 的正确性。如果签名 σ 不合法，S 忽略消息请求，如果 σ 合法，需要做假冒 TPM/TCM 检测，根据撤销列表中 f 验证 $T_3 \stackrel{?}{=} \eta^f$。

(1) 如果找到对应的 f 使得 $T_3=\eta^f$，那么模拟器 S 检查是否存在与 f 对应的 id_i，如果存在这样的 id_i，S 作为主机 H_i 请求可信方 T 对消息 m 签名。如果不存在对应的 id_i，S 检查签名中的消息对 $<\eta, T_3>$ 是否是第一次出现，S 选择一个已经被攻陷的 M_i（该 M_i 还没有成为群成员），以 M_i 的身份请求可信方 T 加入，并且将该 M_i 标记为假冒的，最后以 H_i 的身份对消息 m 签名。

(2) 如果找不到对应的 f，表明对 m 进行签名的平台是假冒的，但还没有放入撤销列表。模拟器 S 必须找出签名来自于哪个平台。模拟器检查签名中的 $<\eta, T_3>$ 在以前是否出现过。

① 如果 $<\eta, T_3>$ 不是第一次出现，S 做以下操作。

- 如果 S 在 Sign 的模拟过程中使用了 $<\eta, T_3>$，但是 S 已经在 Sign 的模拟过程中回答了可信方，那么 S 输出“模拟失败”。S 模拟失败的原因在于攻击方伪造了签名，并且签名中 T_3 是模拟器选择的。因为该签名本质上是对 T_3 的离散对数的零知识证明，因此如果存在这样的攻击方 A，那么就存在另一个攻击方 A'，A' 调用攻击方 A，利用回绕调用(Rewinding)技术能够解决群 G_3 上的 DDH 问题。
- 否则，找到与 $<\eta, T_3>$ 对应的主机 H_i，TPM/TCM M_i，作为主机 H_i 与可信方 T 交互，完成对 m 的签名。

② 如果 $<\eta, T_3>$ 是第一次出现，模拟器 S 为 $<\eta, T_3>$ 找到对应的 TPM/TCM M_i。TPM/TCM 已经被攻陷。

- 如果 SSID$=\perp$，S 任意选择一个没有标记为假冒 TPM/TCM 的 M_i，S 与 T 交互完成对消息 m 的签名。
- 如果 SSID$\neq\perp$，S 选择一个没有标记为假冒 TPM/TCM 的 M_i，如果能够找到这样的 M_i，S 与 T 交互完成对消息 m 的签名。如果找不到，表明攻击方可以成功地伪造签名，S 将模拟失败，但是由引理 16.5 可得，如果攻击方能够成功地伪造签名，那么必定存在一个算法攻破 q-SDH 难题。

引理 16.4 给定 m 的两个签名。

$$\{\eta, T_1, T_2, T_3, c_1, n_t, R_1', R_2', R_3', R_4', s_f', s_t', s_x', s_w', s_{\delta_1}', s_{\delta_2}', H_1\},$$
$$\{\eta, T_1, T_2, T_3, c_2, n_t, R_1', R_2', R_3', R_4', s_f'', s_x'', s_t'', s_w'', s_{\delta_1}'', s_{\delta_2}'', H_2\},$$

使得

$$(s_f', s_x', s_t', s_w', s_{\delta_1}', s_{\delta_2}') \neq (s_f'', s_x'', s_t'', s_w'', s_{\delta_1}'', s_{\delta_2}'')$$

$$c_1 = H_1(H_1(H_1(\eta \| g \| h \| g_1 \| g_2 \| g_3 \| g_T \| Y \| T_1 \| T_2 \| T_3 \| R_1' \| R_2' \| R_3' \| R_4') \| n_t) \| m) \neq c_2$$

$$= H_2(H_2(H_2(\eta \| g \| h \| g_1 \| g_2 \| g_3 \| g_T \| Y \| T_1 \| T_2 \| T_3 \| R_1' \| R_2' \| R_3' \| R_4') \| n_t) \| m)$$

$$R_1' = e(g,g_2)^{s_f'} e(h,Y)^{s_w'} e(h,g_2)^{s_{\delta_1}'+s_t'} e(T_1,g_2)^{s_x'} (e(T_1,Y)/e(g_1,g_2))^{-c_1}$$

$$R_1' = e(g,g_2)^{s_f''} e(h,Y)^{s_w''} e(h,g_2)^{s_{\delta_1}''+s_t''} e(T_1,g_2)^{s_x''} (e(T_1,Y)/e(g_1,g_2))^{-c_2},$$

$$R_2' = T_2^{-c_1} g^{s_w'} h^{s_x'},\quad R_2' = T_2^{-c_2} g^{s_w''} h^{s_x''},\quad R_3' = T_2^{s_x'} g^{s_{\delta_1}'} h^{s_{\delta_2}'},\quad R_3' = T_2^{s_x''} g^{s_{\delta_1}''} h^{s_{\delta_2}''}$$

$$R_4' = T_3^{-c_1} \eta^{s_f'},\quad R_4' = T_3^{-c_2} \eta^{s_f''}$$

那么可以计算出(A,x,t,w,f)满足方程(16-23)。

证明 下面的(A,x,t,w,f)满足引理 16.4。

$$A = \frac{T_1}{h^{\left(\frac{s_w'-s_w''}{c_1-c_2}\right)}},\quad x = \frac{s_x''-s_x'}{c_1-c_2},\quad f = \frac{s_f'-s_f''}{c_1-c_2},\quad w = \frac{s_w'-s_w''}{c_1-c_2},\quad t = \frac{s_t'-s_t''}{c_1-c_2}$$

引理 16.5 在颁发者没有被攻陷的情况下,如果存在攻击方 $\overline{A}$ 运行 Join 协议少于 $q-1$ 次,能够伪造出合法签名$(m,\sigma=(\eta,T_1,T_2,T_3,c,n_t,s_f,s_t,s_x,s_w,s_{\delta_1},s_{\delta_2}))$,那么就存在一个攻击方 $\overline{A}'$能够解决 q-SDH 问题。

证明 假设算法 $\overline{A}'$的输入为 $q+2$ 元组$(g_1,g_2,g_2^{\gamma},g_2^{\gamma^2},\cdots,g_2^{\gamma^q})$,$\psi(g_2)=g_1$,$\overline{A}'$与 $\overline{A}$ 做以下游戏:

(1) $\overline{A}'$随机选择 $\alpha \in_R Z_p$,$\{(a_i,b_i) \in_R (Z_p)^2\}_{i\in[q-1],m\in_R[q-1]}$。

(2) 令 $\omega=\gamma-a_m$,$\overline{A}'$随机选择 $\theta \in_R Z_p$,生成颁发者公钥信息如下。

$$G_2 = g_2^{\left[b_m \prod_{i=1,i\neq m}^{q-1}(\gamma+a_i-a_m)\right]} g_2^{\left[\alpha \prod_{i=1}^{q-1}(\gamma+a_i-a_m)\right]},\quad G_1 = \psi(G_2)$$

$$g = \psi(g_2)^{\left[\prod_{i=1,i\neq m}^{q-1}(\gamma+a_i-a_m)\right]},\quad h = g^{\theta},\quad Y = G_2^{\omega}$$

(3) $\overline{A}'$输出公钥 $pk=(p,G_1,G_2,Y,g,h)$,将 pk 发送给 $\overline{A}$。

(4) 接下来 $\overline{A}'$调用 $\overline{A}$(作为需要加入群的可信计算平台)运行 Join 协议。

① $\overline{A}'$回绕调用(Rewind)$\overline{A}$ 得到 f,t'。

② $\overline{A}'$产生 A,x,t''

$$t'' = (b_i - f)/\theta - t',\quad x = a_i$$

$$A = (G_1 g^f h^{t'+t''})^{1/(\omega+x)} = (G_1 g^{b_i})^{1/(\omega+a_i)}$$

$$= \psi(g_2)^{\left[(b_m+b_i)\prod_{j=1,j\neq m,i}^{q-1}(\gamma+a_j-a_m)\right]} \psi(g_2)^{\left[\alpha \prod_{j=1,j\neq i}^{q-1}(\gamma+a_j-a_m)\right]}$$

$\overline{A}'$发送(A,x,t'')给 $\overline{A}$。

(5) $\overline{A}$ 输出签名对(m,σ),使得验证算法能通过,根据 Forking 引理和引理 16.4,$\overline{A}'$计算得$(A,x,t,f)\in G_1\times(Z_p)^5$,满足等式(16-23)。

其中 $A=(G_1 g^f h^{t'+t''})^{1/(\omega+x)} = (G_1 g^{\theta t+f})^{1/(\omega+x)} = \psi(g_2)^{\left[\frac{\alpha\gamma+(\theta t+f)+b_m}{\gamma-a_m+x}\prod_{j=1,j\neq m}^{q-1}(\gamma+a_j-a_m)\right]}$,将等式进行化简得到 $A=\psi(g_2)^{\sum_{i=0}^{q-1}c_i\gamma^i+(c_q/(\gamma-a_m+x))}$,从以上分析可以得到,对给定的 q-SDH 元组,其解为

$$\left(\left(A/\left(\psi(g_2)^{\sum_{i=0}^{q-1}c_i\gamma^i}\right)\right)^{(1/c_q)},x-a_m\right)$$

3. 模拟器的正确性

最后,证明环境 ε 不能区分自己是运行在现实系统中还是理想系统中,也就是证明现实系统和理想系统中的输出参数是计算不可区分的。在模拟的过程中,S 扮演了现实系统中

的不同角色，模拟器 S 在模拟过程中选择的参数(输出)有一些 s_* 及 T_*，这些参数都是随机选定的，在现实系统中这些参数是由秘密信息经过计算得到的。由于这些参数是统计不可区分的，因此在 Z_p 中也是计算不可区分的。

另外，c 由模拟器随机选择，在现实系统中是 Hash 函数的计算结果，由于是在随机预言机模型下，所以这两者是计算不可区分的。

16.6.3 CF 方案实现考虑

1. 签名长度

在 CF 方案中，如果考虑匿名性保护机制，最后得到的签名 σ 包括了 4 个群 G_1 中的元素，8 个 Z_p 中的元素，假设 $G_1 \neq G_2$，利用文献[48]中定义的椭圆曲线簇，当 $|p| = k = 170$ 时，G_T 和 G_1 中的元素长度分别为 1020b 和 171b，则 CF 方案的签名长度为 2044b。

2. 计算效率

由于指数运算/多指数运算及双线性对运算是最耗时的运算，这里将根据方案中用到的指数运算(多指数运算)和双线性对运算来估算计算开销。下面分别计算方案中加入过程，签名和验证操作的可信计算平台的计算开销(考虑匿名性机制)。其中双线性运算 $e(g,g_2)$、$e(h,Y)$、$e(h,g_2)$、$e(T_1,g_2)=e(A,g_2)\cdot e(h,g_2)^w$ 都是可以预先计算的。

(1) 加入过程：TPM/TCM 做 2 次指数运算。

(2) 签名操作：主机做 5 次指数运算，TPM/TCM 做 4 次指数运算。

(3) 验证操作：4 次多指数运算和 1 次双线性运算。

3. 与其他方案的比较

这里将 CF 方案与 BCC 方案和 HS 方案进行比较，具体的性能指标在表 16.3 列出，其中“BM”表示双线性运算，“ME”表示指数运算(多指数运算)，“SC”表示模平方运算，“MC”表示乘法运算。并且由于在计算过程中 TPM/TCM 的计算量是一个非常重要的性能指标，因此将比较在签名过程中 TPM/TCM 的计算量。对于指数运算，将参照文献[49]给出的方法估算计算开销，对于某个指数运算，设 m_1 是指数的二进制表示的比特长度，m_2 是指数的二进制表示中 1 的个数。那么该指数运算的计算开销可以估算为 m_1 次模平方运算和 m_2 次乘法运算。

在 HS 方案[50]中，安全参数的一般取值为

$$l_n = 2048,\quad \alpha = 9/8,\quad X = 2^{792},\quad Y = 2^{520},$$
$$l_s = 540,\quad l_b = 300,\quad l_c = 160$$

那么其长度至少为 7614b。

而在 BCC 方案中，安全参数的一般取值为

$$l_n = 2048,\quad l_f = 104,\quad l_e = 368,\quad l'_e = 120,\quad l_v = 2536,$$
$$l_\phi = 80,\quad l_H = 160,\quad l_r = 80,\quad l_\Gamma = 1632,\quad l_\rho = 208$$

其长度至少为 20555b。

从表 16.3 可以看出，CF 方案与其他的直接匿名证明方案相比，签名长度大大缩短，能有效地节省传输带宽，更重要的是由于 CF 方案可以采用椭圆曲线来实现，因此指数运算的效率要比其他方案高，同时最耗时的双线性对运算可以在主机上执行，为新一代基于 ECC

算法的 TPM/TCM 提供了一种隐私性保护解决方案。

表 16.3 直接匿名证明方案的性能比较

方案名	签名长度	签名过程总计算量（TPM/TCM 的计算量）	加入(Join)过程计算量	验证过程计算量	基于的假设
BCC 方案	20555b	8ME+0BM (16186SC+8093MC)	4ME+0BM	4ME+0BM	强 RSA,DDH
HS 方案	7614b	3ME+0BM (2352SC+958MC)	5ME+0BM	3ME+0BM	强 RSA,DDH
CF 方案	2044b	9ME+0BM (855SC+425MC)	4ME+0BM	4ME+1BM	q-SDH,DDH

4. 椭圆曲线选择

超椭圆曲线是具有附加代数结构的特殊曲线，在密码算法中，一直以来都避免使用这类曲线，因为附加的代数结构使得这类曲线易于遭受一些特殊的攻击。但是，一般标准的椭圆曲线密码学系统（如 ElGamal 加密算法或者 ECDSA 签名算法）可以使用随机产生的椭圆曲线实现，实现基于双线性对的密码系统需要的椭圆曲线必须具有一定的性质，不能够随机产生，而超椭圆曲线可以实现双线性对。

下面选择合适的椭圆曲线。对于群 G_1，G_2，G_T 及其所对应的双线性映射，可以使用文献[48]给出的椭圆曲线，如可以使用以下的有限域 F_{p^2}（$p\equiv 3 \bmod 4$）上的超椭圆曲线：

$$E: y^2 = x^3 + x$$

利用以上椭圆曲线实现的双线性对基于 F_{p^2} 上的离散对数问题，取 p 为 512b，其安全强度相当于解决 1024b 的 RSA 问题。

16.7 小结

可信计算平台技术目前已成为计算机安全技术最主要的发展趋势之一，而远程证明是可信计算平台提供的一大特色和核心功能，是目前可信计算领域最为热门的研究方向之一。信息安全国家重点实验室可信计算项目组针对可信计算中的一些核心问题进行了深入系统研究，并研制了一套可信计算测评原型系统，在可信计算信任链构建、远程证明、测评方法等方面取得了一批自主创新成果，部分工作已在文献[52]～[56]中做了较为详细的总结。本章主要介绍了我们自己的一些工作，对其他一些有代表性的工作也做了一些综述和对比介绍，感兴趣的读者可从文中介绍的线索参阅相关文献，中文综述文献可参阅文献[57]。陈晓峰博士和秦宇博士也参加了本章的写作，作者在此表示衷心的感谢。

另外，值得一提的是，在本章的写作过程中交替使用了 TPM 和 TCM 这两个核心术语，有的方案使用了其中一个，有的方案二者兼用，其实这不是本质的，不管是 TPM 还是 TCM，只要提供适当的密码运算功能，这些方案都可以使用。

参考文献

[1] Trusted Computing Group. www.trustedcomputinggroup.org.
[2] Trusted Computing Group, TPM Main Part 1, Design Principles Specification, Version 1.2 Revision

62, 2 October 2003 Published, https://www.trustedcomputinggroup.org/home.
[3] 国家商用密码管理公告(第 13 号):可信计算密码支撑平台功能与接口规范. 国家商用密码管理办公室. http://www.oscca.gov.cn/.
[4] Sailer R, Zhang X L, Jaeger T, Van Doorn L. Design and Implementation of a TCG-based Integrity Measurement Architecture. 13th Usenix Security Symposium, San Diego, California, August, 2004.
[5] Sailer R, Van Doorn L, James P W. The Role of TPM in Enterprise Security. Datenschutz und Datensicherheit (DuD), September, 2004.
[6] Poritz J, Schunter M, Van Herreweghen E. Property Attestation-Scalable and Privacy-friendly Security Assessment of Peer Computers, IBM Research Report RZ 3548, October 5, 2004.
[7] Sadeghi A, Stüble C. Property-based Attestation for Computing Platforms: Caring about properties, not mechanisms, New Security Paradigms Workshop, September 2004.
[8] Chen L Q, Landfermann R, Löhr H. A protocol for property-based attestation. In Proceedings of the first ACM workshop on Scalable trusted computing, Nova Scotia Canada, ACM Press: 7～16, 2006.
[9] Kuehn U, Selhorst M, Stueble C. Realizing property-based attestation and sealing with commonly available hard-and software. In Proceedings of the 2007 ACM workshop on Scalable trusted computing, Alexandria, Virginia, USA, November 2007.
[10] Kühn U, Kursawe K, Lucks S. Ahmad-Reza Sadeghi & Christian Stüble: Secure Data Management in Trusted Computing. In: Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES), Springer 2005, LNCS 3659, 324～338.
[11] TCG Infrastructure Working Group Reference Architecture for Interoperability (Part I) Specification, Version 1.0, Revision 1, 16 June 2005, https://www.trustedcomputinggroup.org/home.
[12] TCG Trusted Network Connect, TNC Architecture for Interoperability Specification Version 1.0 Revision 4,3 May 2005 Published, https://www.trustedcomputinggroup.org/home.
[13] Trusted Computing Group, TLS Extensions for Attestation, Specification Version 1.0, Revision 0.8, July 2004, Work in Progress.
[14] Gasmi Y, Sadeghi A R, Stewin P. Beyond secure channels. In Proceedings of the 2007 ACM Workshop on Scalable Trusted Computing. Alexandria, Virginia, USA, November 2007.
[15] Intel. Virtualization Technology, http://www.intel.com/technology/computing/vptech/, 2005.
[16] Barham P, Dragovic B, Fraser K. Xen and the Art of Virtualization. In Proceedings of the 19th ACM Symposium on Operating Systems Principles, Bolton Landing, NY, Oct. 2003.
[17] Hand S, Warfield A, Fraser K. Are Virtual Machine Monitors Microkernels Done Right. In the Proceedings of the Tenth Workshop on Hot Topics in Operating Systems (HotOS-X), June 2005.
[18] Intel. LaGrande Technology Architectural Overview. http://download.intel.com/technology/security/downloads/LT_Arch_Overview.pdf.
[19] Microsoft. Microsoft NGSCB home page, http://www.microsoft.com/resources/ngscb, 2003.
[20] Bellare M, Rogaway P. Random oracles are practical: A paradigm for designing efficient protocols. In Proceedings of the First Annual Conference on Computer and Communications Security, 1993.
[21] TCGInfrastructure Working Group (IWG). TCG Infrastructure Workgroup Subject Key Attestation Evidence Extension, Specification Version 1.0 Revision 7, June 2005. https://www.trustedcomputinggroup.org/specs/IWG/IWG_SKAE_Extension_1-00.pdf.
[22] Ralph C. Merkle. A certified digital signature. In Proceedings on Advances in Cryptology 1989. Santa Barbara, California, USA.

[23] Van Dijk M, Sarmenta L F G, O'Donnell C W. Offline untrusted storage with immediate detection of forking and replay attacks. In Proceedings of the 2007 ACM Workshop on Scalable Trusted Computing(STC'07), Alexandria, Virginia, USA, November 02, 2007.

[24] Buldas A, Laud P, Lipmaa H. Accountable Certificate Management using Undeniable Attestations. In Proceedings of the 7th ACM Conference on Computer and Communications Security, pages 9～17, 2002.

[25] 冯登国,秦宇,可信计算环境证明方法研究,计算机学报,Vol. 31,No. 9,2008,1640～1652.

[26] Camenisch J,Stadler M. Efficient group signature schemes for large groups. In B. Kaliski,editor, Advances in Cryptology—CRYPTO' 97, volume 1296 of LNCS, pages 410 ～ 424. SpringerVerlag, 1997.

[27] Fiat A,Shamir A. How to prove yourself: Practical solutions to identification and signature problems. In CRYPTO'86, vol. 263 of LNCS, 186～194.

[28] Chase M, Lysyanskaya A. On Signatures of Knowledge. In Advances in Cryptology—CRYPTO 2006.

[29] Camenisch J,Lysyanskaya A. Signature schemes in Cryptology—CRYPTO' 04,Vol. 3152,56～72,2004.

[30] Pedersen T P. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, Advances in Cryptology—CRYPTO'91, volume 576 of LNCS, pages 129～140. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1992.

[31] 冯登国,秦宇.一个基于 TCM 的属性证明协议,中国科学:信息科学,2010,40:189～199.

[32] Ernest F. Brickell, Camenisch J. Chen L Q. Direct anonymous attestation. ACM Conference on Computer and Communications Security 2004:132～145.

[33] Camenisch J. Better Privacy for Trusted Computing Platforms:(Extended Abstract). ESORICS 2004:73～88.

[34] Camenisch J, Groth J. Group signatures: Better efficiency and new theoretical aspects. In C. Blundo and S. Cimato, editors, Proceedings of Forth International Conference on Security in Communication Networks, SCN 2004, volume 3352 of LNCS, pages 122～135. Springer, 2005.

[35] Camenisch J,Lysyanskaya A. A signature scheme with efficient protocols. In SCN 2002, vol. 2576 of LNCS, 268～289. Springer Verlag, 2003.

[36] Smyth B, Chen L, Ryan M. Direct anonymous attestation (DAA): ensuring privacy with corrupt administrators. In F. Stajano, editor, Proceedings of Fourth European Workshop on Security and Privacy in Ad hoc and Sensor Networks (ESAS 2007), volume 4572 of LNCS,pages 218～231. Springer-Verlag, 2007.

[37] Backes M, Maffei M, Unruh D. Zero-knowledge in the applied pi-calculus and automated Verification of the direct anonymous attestaion protocol. Cryptology ePrint Archive, Report 2007/289, 2007. http://eprint. iacr. org/.

[38] Camenisch J. Protecting (Anonymous) Credentials with the Trusted Computing Group's TPM V1. 2. SEC 2006:135～147.

[39] Leung A,Mitchell C J. Ninja: Non identity based, privacy preserving authentication for ubiquitous environments. In Proceedings of 9th International Conference on Ubiquitous Computing, volume 4717 of LNCS, 73～90. Springer-Verlag, 2007.

[40] Brickell E,Chen L Q,Li J T. Simplified Security Notions of Direct Anonymous Attestation and a Concrete Scheme from Pairings,In Conference on Trusted Computing(TRUST 2008), Villach,

Austria, March 2008.
[41] 陈小峰,冯登国. 一种多信任域内的直接匿名证明方案,计算机学报,2008,31(7):1122~1130.
[42] Camenisch J, Anna Lysyanskaya A. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. CRYPTO 2002: 61~76.
[43] Chen X F, Feng D G. Direct Anonymous Attestation for Next Generation TPM. Journal of Computer, Vol. 3, No. 12, December 2008, 43~50.
[44] Boneh D, Boyen X, Shacham H. Short Group Signatures. CRYPTO 2004: 41~55.
[45] Boneh D, Shacham H. Group signatures with verifier-local revocation. ACM Conference on Computer and Communications Security 2004: 168~177.
[46] Canetti R. Studies in Secure Multiparty Computation and Applications. PhD thesis, Weizmann Institute of Science, Rehovot 76100, Israel, June 1995.
[47] Pfitzmann B, Waidner M. Composition and integrity preservation of secure reactive systems. In Proc. 7th ACM Conference on Computer and Communications Security, pages 245~254. ACM Press, Nov. 2000.
[48] Miyaji A, Nakabayashi M, Takanov S. New explicit conditions of elliptic curve traces for FR-reduction. IEICE Trans. E85-A(2), 481~484, 2002.
[49] Menezes A J, Oorschot P C, Vanstone S A. Handbook of Applied Cryptography, pages 613~619. CRC Press, Inc, 1997.
[50] Ge H, Stephen R. A Direct Anpnymous Attestation Scheme for Embedded Devices. Public key Cryptography2007:16~30.
[51] Chen X F, Feng D G. A New Direct Anonymous Attestation Scheme from Bilinear Maps. ICYCS 2008: 2308~2313.
[52] 陈小峰. 可信平台模块分析与测试技术研究. 中国科学院软件研究所博士学位论文,2008.
[53] 秦宇. 可信虚拟平台安全机制研究,中国科学院软件研究所博士学位论文,2008.
[54] Aimin Y, Feng D G, Liu R. TBDRM: A TPM-Based Secure DRM Architecture. CSE (2) 2009: 671~677.
[55] Qin Y, Feng D G, Xu Z. An Anonymous Property-Based Attestation Protocol from Bilinear Maps. CSE (2) 2009: 732~738.
[56] Zhang Y, Zhai Z D, Feng D G. Bring Efficient Connotation Expressible Policies to Trust Management. ICICS 2009: 396~410.
[57] 冯登国. 可信计算技术研究进展,中国密码学发展报告 2008. 北京:电子工业出版社,2009.